كتاب الجافا

سؤال وجواب



فكرة و إعداد
سعد رضا العنزي

تم اعداد هذا الكتاب مع مجموعة من المبرمجين في جروب مبرمجي الجافا و الأندرويد وتم ذكر اسمائهم في آخر هذا الكتاب .. وهذا الكتاب تجميع لبعض الأسئلة المهمة في الجافا والتي تفيد المبتدئين والمتقدمين في مراجعة بعض المفاهيم والمصطلحات المهمه في لغة الجافا .

بسم الله الرحمن الرحيم



أسأل الله أن يتقبل هذا العمل ويجعله خالص لوجه الكريم ، وأشكر كل من ساهم في الكتاب ولو بسؤال واحد ، وكذلك لا أنسى أشكر منتديات الفريق العربي للبرمجة لما يقدموه من إثراء للمحتوى العربي (وقد تم أخذ بعض الأسئلة منهم) .

...

ولا تنسوني وجميع المساهمين من صالح دعائكم في.. ظهر الغيب

المحتويات

الصفحة

الوحدة الأولى : الأساسيات

الوراثة (Inheritance)

Databases

Graphics 2D

مشاريع جاهزة

الأساسيات

س١ : ماهي لغة جافا وما هي بداياتها ؟

جافا (Java) هي عبارة عن لغة برمجة ابتكرها جيمس جوسلينج في عام ١٩٩٢ م أثناء عمله في مختبرات شركة صن ميكروسيستمز، وذلك لاستخدامها بمثابة العقل المفكر المستخدم لتشغيل الأجهزة التطبيقية الذكية مثل التيلفزيون التفاعلي، وقد كانت لغة الجافا تطويرًا للغة السي بلاس بلاس (C++)، وعند ولادتها أطلق عليها مبتكرها "أواك" بمعنى شجرة السنديان؛ وهي الشجرة التي كان يراها من نافذة مكتبه وهو يعمل في مختبرات صن ميكروسيستمز، ثم تغير الاسم إلى جافا، وهذا الاسم (على غير العادة في تسمية لغات البرمجة) ليس الحروف الأولى من كلمات جملة معينة أو تعبيرا بمعنى معين، ولكنه مجرد اسم وضعه مطوّروا هذه اللغة لينافس الأسماء الأخرى، وهى احدى لغات البرمجة، أي مجموعة من الأوامر والتعليمات التي تعطى للحاسب في صورة برنامج مكتوب بلغة برمجة معينة بواسطة معالج نصوص، ويتكون مصدر البرنامج من عدة سطور وكل سطر يعتبر جملة، ويتعامل الحاسب مع كل جملة بترتيب معين لإنجاز الأمر الذي صمم البرنامج لتحقيقه.

س ٢ : ماهي مميزات لغة جافا ؟

- السهولة.
- داعمة و موجهة للكيانات.
- سهولة الحصول عليها.
- آمنة.
- قابلة للنقل و التنفيذ.
- إضافة الحركة والصوت إلى صفحات الويب.
- كتابة الألعاب والبرامج المساعدة.
- إنشاء برامج ذات واجهة مستخدم رسومية.
- تصميم برمجيات تستفيد من كل مميزات الأنترنت. توفر لغة الجافا بيئة تفاعلية عبر الشبكة العنكبوتية وبالتالي تستعمل لكتابة برامج تعليمية للإنترنت عبر برمجيات المحاكاة الحاسوبية للتجارب العلمية وبرمجيات الفصول الافتراضية للتعليم الإلكتروني والتعليم عن بعد. لا تنحصر فاعلية الجافا في الشبكة

العنكبوتية فقط بل تمكننا من إنشاء برامج للاستعمال الشخصي والمهني، هذه البرامج تنجز عبر جملة من البرامج التي تسهل كتابة الأوامر كبرنامج NetBeans وEclipse.

س ٣ : ما إصدارات الجافا ؟

Java SE

الإصدار القياسي من الجافا بالإنجليزية (Java Standard Edition) هو الجزء الخاص ببرمجة برامج سطح المكتب (StandAlone applications).

Java EE

اصدار الإنتربرايس بالإنجليزية (Java Enterprise Edition)هو يختلف عن الإصدار القياسي في أنه خاص ببرمجة الخوادم والتطبيقات الموزعة وبرمجة الويب.

Java ME

اصدار المايكرو بالإنجليزية (Java Micro Edition) وهو إصدار خاص بالبرمجة على الهواتف المحمولة والأجهزة المحدودة في المصادر عموما وتعتبر الخيار الأول لمبرمجي ألعاب الهواتف المحمولة.

Java Fx

تقنيّة أخرى اصدرتها صن ميكروسيستمز لتطوير تطبيقات من نوع "RIA" أي "Rich Internet Applications".

س٤ : كم يتطلب الوقت من ممارسة وتعلم الجافا حتى تصبح محترف ؟

الجافا لغة هائلة وكبيرة وطبعا المدة تكون حسب همة الشخص وارادته ولكي تصبح محترف في اللغة كوقت تقريبي يحتاج لك ثلاثة إلى خمسة سنين .

س٥: ما هو مدى انتشار الجافا خصوصا في الوطن العربي؟

في الحقيقة الجافا منتشرة في الدول المتقدمة بصورة كبيرة جدا و خصوصاً في (امريكا وبريطانيا) و لكن مع الاسف فهي عالمنا العربي قليلة الانتشار و أن كانت الجامعات أدخلتها و لكن منذ مدة بسيطة نسبياً.

س٦ : ماهي الكلمات المحجوزة في لغة جافا اي الكلمات التي لا يسمح باستخدامها إلا للغرض الذي انشأت من اجله ؟

Java Keywords الكلمات المحجوزة في لغة الجافا		
abstract	finally	public
boolean	float	return
break	for	short
byte	if	static
case	implements	super
catch	import	switch
char	instanceof	synchronized
class	int	this
continue	interface	throw
default	long	throws
do	native	transient
double	new	true
else	null	try
extends	package	void
false	private	volatile
final	protected	while

س٧ : كيف يمكن إعادة تسمية class معين من على أحد الـ IDE ؟

تضغط على الكلاس بالزر الأيمن ثم تضغط من القائمة على Refactor ثم تختار rename وسوف يتم تحديث الاسم في الكود تلقائيا دون تدخل منك

س٨ : ما هو المقصود بـ variable وvalue في جافا؟

Variable: هو المتغير .

Value: هو اعطاء او اسناد قيمه لذلك المتغير .

س٩ : ما هو الكونستراكتر (Constructor) ؟

يستخدم الكونستراكتر لإعطاء القيم الأولية للأوبجكت في الكلاس ويكون له نفس إسم الكلاس تماماً

هناك اختلاف ما بين الكونستراكتر والميثود في الآتي :

.الكنوستراكتر : يستخدم لإعطاء القيم الابتدائية للأوبجكت عند خلقه .

. الميثود : تعطي قيم للأوبجكت ولكن عند استدعائها فقط .

الكونستراكتر في الكلاس لايورث للسب كلاس ويعرف على أنه public

حتى تسهل عملية خلق الأوبجكت منه خارج الكلاس نفسه .

The KeyWord "THIS"

تستخدم بغرض استدعاء الأوبجكت .

key word "SUPER"

تستخدم في كونستراكتر السب كلاس ذلك لتقوم بوراثة جميع المتغيرات والقيم المعرفة في السوبر كلاس .

س١٠ : كيف اقوم بإنشاء كائن من كلاس بأشكال مختلفة (أي عن طريق ادراج بارامترات مختلفة) ؟

يكون هذا عن طريق انشاء عدد من الكونستراكتر لنفس الكلاس وذلك حسب الحاجة له

مثال :

انشاء الكونستراكتر الافتراضي {}()public constructor

انشاء الكونستراكتر الاول بباراميتر واحد {}(public constructor(int a

انشاء الكونستراكتر الثاني بباراميترين ... {}(public constructor(int a,int b
الخ .

س١١ : ما معنى البريفايت (private) و البروتكتد (protected) و البابلك (public) بالنسبة للمتغيرات ؟؟

كل متغير في لغة جافا ويتميز بخاصية من الخواص الثلاثة المذكورة اعلاه والتي تسمى بـ(acces modefires)، وهذه الخواص هي التي تسمح بإمكانية الوصول الى المتغير من عدمها .

برايفت (private): و هي خاصية تشفير متكامل للمتغير حيث يمنع منعا باتا الوصول اليه من خارج الكلاس المعرف بداخلها مباشرة (أي عن طريق اسمه) .

بروتكتد (protected) : وهي خاصية اقل درجة من البريفايت حيث تسمح للكلاسات الوارثة من الكلاس الاصلي الذي عرف المتغير بداخله من الوصول اليه بسهولة (أي عن طريق الاسم ، اسم المتغير مباشرة) .

بابليك (public) : هي الخاصية الثالثة وهي ألين الخواص ، حيث تسمح لك بالوصول للمتغير من أي مكان شئت .

ملاحظة : اذا لم نقم بإدراج أي خاصية من هذه الخواص للمتغير فان لغة جافا تقوم بمنحه خاصية Default Access Modifier افتراضيا .

والـ Default Access Modifier :

نوع رابع لـAccess Modifier لكن غير مشهوره جدا

المتغير أو الدالة التي لما تسبقها أي كلمة من الثلاث السابقة private أو protected أو public فهي تعتبر Default .

و التعامل معها محصور داخل الكلاسات التي داخل الـ pakage .

س١٢ :كيف أخزن قيمة في متغير ؟

يمكن تحديد قيمة أولية للمتغير عندما نقوم بإنشائه ونستطيع أيضا تحديد قيمة للمتغير في أي موضع تالي ضمن البرنامج وتستعمل علامة = لتحديد قيمة أولية للمتغير عند إنشائه ويمكن أن نأخذ مثال

```
int i = 500;
```

نلاحظ أننا أنشأنا متغيرا i

هذا المتغير يحتوي على القيمة الإبتدائية ٥٠٠ ، وهكذا وبنفس الطريقة يمكننا إعداد المتغيرات المخزنة للأعداد ..

ويختلف الأمر بالنسبة للمتغيرات المخزنة لسلسلة نصية حيث يجب وضع علامات اقتباس حول القيمة المراد تخزينها

```
String studentName = "Hamza ";
```

س١٣ : هل يمكن إعداد أحد المتغيرات ليساوي قيمة متغير آخر من نفس النوع ؟

نعم يمكننا إعداد متغير ليساوي قيمة متغير آخر من نفس النوع كما نرى في المثال التالي

```
int i = 500;

int k= i ;
```

س ١٤ : لكن ماذا يحدث إذا لم أحدد قيمة ابتدائية للمتغير؟

هنا يحدث خطأ ويظهر مترجم الجافا رسالة خطأ ولذلك يجب تحديد قيمة أولية للمتغير

س ١٥ : هل هناك قواعد معينة لتسمية المتغيرات ؟

نعم هناك قواعد يجب مراعاتها عند تسمية المتغيرات حيث تبدأ أسماء المتغيرات بحرف إنجليزي أو علامة الدولار $ أو علامة التسطير (-) ويكون باقي الإسم حرف أو عدد وبشرط عدم وضع مسافات فارغة ولا يمكننا إستعمال أيضا علامات الترقيم حيث يظهر مترجم الجافا رسالة خطأ إذا كان في إسم المتغير مسافات فارغة أو علامات ترقيم واذا استعملت حرفا كبيرا في اسم متغير فانه يجب استعماله بنفس الطريقة في كل البرنامج على سبيل المثال :

studentName هذا اسم لمتغير لن يكون نفس هذا

StudentName تغيير الحرف الأول من صغير الى كبير في نفس البرنامج لاسم المتغير يؤدي الى خطأ عند الترجمة

ولذلك يجب أن يكون اسم المتغير دالا على الغرض من استعمال هذا المتغير والحرف الأول من اسمه يجب أن يكون صغيرا وفي حالة تكون اسم المتغير من أكثر من كلمة يكون الحرف الأول من الكلمة الأولى صغيرا وباقي الكلمات تبدأ بحروف كبيرة وبدون فواصل أو علامات ترقيم

س ١٦ : ماذا عن العمليات الرياضية على المتغيرات ؟

كما قلنا سابقا تطلب البرمجة عموما قدرات رياضية معينة وفكر رياضي في المبرمج وعلى الرغم من قيام الحاسب بكل العمليات الرياضية الا أنه يريد تعليمات وأوامر من المبرمج حتى يقوم بهكذا عمليات وتسمى الأوامر التي تعطيها للحاسب والتي تحتاج الى عمليات رياضية تعابير ويمكننا

استخدام هذه التعابير للقيام بعدة مهام منها تغيير قيمة متغير واستعمال المعادلات في البرنامج وتسجيل عدد مرات حدوث عمل ما في البرنامج وتستعمل هذه التعابير الجمع والطرح والضرب والقسمة وباقي القسمة

يعني هذا أننا رجعنا الى المرحلة الابتدائية والى مدرس الحساب اليس كذلك

ما أجمل العودة الى ذكريات الطفولة والمرحلة الأولى وتلك الأيام الخوالي حيث البراءة والأحلام الوردية ـ ما علينا ـ نرجع بالذاكرة الى أيام مدرس الرياضيات في المرحلة الإعدادية ونسترجع العمليات الأربع الشهيرة في الحساب الجمع والطرح والضرب والقسمة وعلاماتها الأربع المعروفة +

و - و * و / هذه الرموز تسمى مؤثرات وتستعمل هذه المؤثرات في برامج الجافا لإجراء العمليات الرياضية على الأرقام طبعا يراودك تساؤل عن الإختلاف بين * و × وبين / و ÷ ولكن كل أمر نتفق فيه على قواعد من البداية يسهل الأمور وهذا بمثابة الدستور الذي نمشي على منهاجه

معاملات (مؤثرات) operators

نستعمل العلامة + لإجراء عملية الجمع

نستعمل العلامة - لتعبير الطرح

يستعمل تعبير الضرب العلامة * في برامج الجافا

يستعمل تعبير القسمة العلامة / في برامج الجافا

نستعمل المؤثر % لتحديد باقي القسمة

نستعمل المؤثر ++ لزيادة القيمة المخزنة في المتغير بمقدار واحد

يستخدم المعامل -- لإنقاص قيمة المتغير بمقدار واحد

س١٧ : ألا ترى أن الأمور قد بدأت في التعقيد يعني إذا كان هناك أكثر من عملية حسابية في البرنامج الا ينبغي أن أعرف الترتيب الذي يتبعه الحاسب لإنجاز هذه العمليات حتى يمكنني ضبط قيمة المتغير ؟

أحسنت هذا موضوع هام جدا ونسترجع من الدراسة في المرحلة الثانوية كيف كان مدرس الرياضيات يركز على هذا الأمر ومن يتوزع على الآخر ومن يسبق من في هذه العمليات الرياضية وفي برامج الحاسب تتم هذه العمليات حسب الترتيب التالي

يتم أولا الزيادة والنقص بمقدار واحد

يلي ذلك الضرب والقسمة وباقي القسمة

يأتي بعد ذلك الجمع والطرح

ثم المقارنة

وتستخدم العلامة = لضبط قيمة المتغير

س ١٨: هل يرجع المتغير بوليان الذي يستعمل لتخزين قيمتين فقط هما ـ صواب، خطأ ؟

نعم يا صديق الجملة اذا تعمل بنفس المبدأ حيث تختبر الشرط من حيث الصواب أو الخطأ ولا تقوم بالعمل إلا إذا كان الشرط صحيحا

مثال بسيط يوضح الأمور لو سمحت ؟

```
if (hour < 12)
System.out.println("Good morning.");
```

ونلاحظ هنا أن جواب الشرط يرتبط بفعل الشرط بمعنى أنه اذا لم يتحقق الشرط فانه لن يحدث شيء

ولكن في بعض الحالات أريد أن أقارن بين قيمتين فماذا أفعل ؟

اذا أردت أن تختبر هل قيمة معينة تساوي أخرى أو أقل منها يمكنك استخدام المؤثر أصغر من

=< أو يساوي

يستخدم لاختبار أكبر من أو يساوي <=

<= & >=

هل يمكنني إختبار المساواة فقط ؟

نعم يمكنك اختبار ما اذا كان متغير ما يساوي قيمة معينة أو لا وهل متغير ما يساوي متغير آخر باستعمال المؤثر == ونلاحظ أنه يتكون من علامتي يساوي ونؤكد على أنه لا تستعمل علامتي التساوي الا في الجمل الشرطية

طيب واختبار عدم المساواة كيف يكون ؟

نستعمل المؤثر =! لاختبار عدم المساواة

وهل نستخدم المؤثرين السابقين لكل أنواع المتغيرات ؟

نستخدم المؤثرين ==   &   =! لكل أنواع المتغيرات ما عدا المتغير استرينج الخاص بسلسلة الحروف وقد تكلمنا في الدرس السادس عن هذا الموضوع وأشرنا في حينه الى استعمال النهج ايكوال لاختبار تساوي سلسلتي حروف

في جميع الأمثلة السابقة تجعل جواب الشرط أمرا واحدا وهو النهج

println ( )

ولكننا في عديد من الحالات قد نرغب في انجاز أكثر من مهمة كنتيجة لفعل الشرط فماذا نفعل ؟

كلامك صحيح ياصديقي معك حق وأبشرك أنك الآن قد بدأت تنتهج النهج الصحيح في البرمجة وهذه الملحوظة تأخذنا الى الغوص في بحر البرمجة لنبحث عن الدر في صدفاته

لكي تنجز أكثر من مهمة كنتيجة لتحقق الشرط عليك أن تنشأ جمل كتلية بواسطة { } الحاصرتين

بالمناسبة الجمل الكتلية هي جمل مرتبة في اطار مجموعة

main( )

وهذه الكتلة تبدأ بحاصرة الفتح { وتنتهي بحاصرة الغلق } اذاً نستعمل الجمل الكتلية مع فعل الشرط لكي نجعل الحاسب يقوم بإداء عدة مهام كنتيجة لتحقق الجملة الشرطية

```
if (minute != 0) {
System.out.print(" " + minute + " ");
System.out.print( (minute != 1) ? "minutes" : "minute");
System.out.print(" past");
}
```

س ١٩: في بعض الأحيان نريد من الحاسب انجاز عمل ما اذا كان الشرط صحيحا والقيام بعمل آخر اذا كان هذا الشرط غير صحيح فماذا نفعل أستاذي العزيز في هذه المشكلة ؟

لكي تفعل ذلك عليك استعمال الجملة

if مع الجملة else

```
if (hour < 12)
System.out.println("Good morning.\n");
else if (hour < 17)
System.out.println("Good afternoon.\n");
else
System.out.println("Good evening.\n");
```

```
if ( grade = = 'A')
System.out.println(" ناجح بامتياز ");
```

```
else if ( grade = = 'B')
System.out.println(" ناجح بتقدير جيد جدا  ");
else if ( grade = = 'C')
System.out.println("ناجح بتقدير جيد   ");
else if ( grade = = 'D')
System.out.println("ناجح بتقدير مقبول   ");
else
System.out.println(" راسب وباق للاعادة  ");
```

س ٢٠: الحالات السابقة فيها شرطان فقط باستثناء المثال أعلاه الا يوجد طريقة أخرى للتعامل مع شروط مختلفة متنوعة

يوجد طريقة أخرى لاختبار مجموعة متنوعة من الشروط والاستجابة لكل منها منفردا هذه الطريقة هي استعمال الجملة

switch

```
switch (month) {
case (1):
System.out.print("January");
break;
case (2):
System.out.print("February");
break;
case (3):
System.out.print("March");
```

```
break;
case (4):
System.out.print("April");
break;
case (5):
System.out.print("May");
break;
case (6):
System.out.print("June");
break;
case (7):
System.out.print("July");
break;
case (8):
System.out.print("August");
break;
case (9):
System.out.print("September");
break;
case (10):
System.out.print("October");
break;
case (11):
System.out.print("November");
```

```
break;
case (12):
System.out.print("December");
}
```

يحدد السطر الأول من الجملة سويتش المتغير المراد اختباره وهو هنا متغير الشهر ثم نتستعمل بعد ذلك الحاصرتين لتكوين جملة كتلية

الجملة case

تختبر هذه الجملة قيمة متغير الاختبار المحدد في الجملة سويتش مقارنة بقيمة معينة

س ٢١: ما هي أنواع الMethods (الدوال) في الجافا ؟

هناك نوعان من الميثودز :

Getter : وتسمى accessors methods ولا تغير شيئا من حالة الأوبجكت وتكون معها كلمة return ملازمة لها .

مثال : frog.getPostion() لاترجع أي قيم.

Setter : وتسمى Mutater methods تغير من حالة الاوبجكت ولاتوجد معها كلمة return مثال : frog.setColour(RED) غيرت الاوبجكت فروغ للون الاحمر.

س ٢٢: ما هي الـ packages ؟

هي مجموعة من الكلاسات يتم استدعائها عند الحاجة إليها. وتستخدم كلمة import لاستدعائها ، وتقوم باستدعاء جميع الكلاسات ذات العلاقة في الباكج عند وضع علامة النجمة (*) .

هناك نوعين من الباكج في الجاوا :

١- standered java backages --------> كلاسات جاهزة للاستخدام.

٢- developer java packages --------> تكتب او تطور بواسطة المبرمج نفسه.

س ٢٣: ما هو الفرق بين ال overriding & overloading ؟

overriding : تعني أن نفس الميثود بنفس الأسم في السوبر كلاس والسب كلاس لكننا نعدل عليها في السب كلاس حسب الأمر المراد تغييره أو تعديله .

overloading :نفس الميثود في نفس الكلاس لكنها تختلف في عدد الباراميرات .

س ٢٤: ما هي أنواع المتغيرات في الجاوا ؟

primitive data type >> مجموعة من القيم وتنقسم الي انواعنا المعروفة ، مثل :

int, double, byte, etc

refrenace type >> تشير إلى اوبجكت مثل : String , Array .

س ٢٥: ما هو مصطلح الـ escape sequence ؟

هو مصطلح يعني استخدام علامة الباك سلاش (\) ليغير مفعول ما بعد الحروف في لغة الجافا فمثلا :

'\n' تعني سطر جديد.

س ٢٦: ما معنى استخدام static في الـ methods & variables ؟

عندما يحتوي الكلاس علي أي من static methods OR variables تعني انها تحتوي على نسخة واحدة فقط في الكلاس لكل الاوبجكتز .

يتم استخدام static variables مع الثوابت او عند تعريف variable واحد في كل الكلاسات .

س ٢٧: ما هي باكج *.java.lang ؟

هذه الباكج يتم استخدامها في جميع كلاسات الجافا لذلك ليس هناك داعي للإستدعائها في كل برنامج ،، لأنه يتم استدعائها اتوماتيكيا .

س ٢٨: ما هو الفرق بين (&) و (&&) في جمل المقارنة ؟

الفرق بينهما:

الحالة الأولى (&) :

في هذه الحالة يقوم المترجم (الكومبيلر) بالنظر إلى المدخل الأول فإذا كان خاطئا فإنه لا يفحص المدخل الثاني .

مثاله:

```
public void and()
  {
    int a = 10 ;
    int b = 5 ;
    if ((a == 22) & (++b == 5))
     System.out.println("case one");

   System.out.println (b);
  }
```

بالطبع لن يقوم المترجم بطباعة ("case one") لأن أحد الشرطين لم يتحقق ولكن عندما يطبع b فإنه سيطبعها كما هي دون أن يزيد عليها لأنه لم يفحص المدخل الثاني ..

لذا سيطبع لنا ٥ ..

الحالة الثانية (&&) :

أما في هذه الحالة فإن المترجم ينظر إلى المدخل الأول والثاني معا حتى لو كان المدخل الأول خاطئاً ..

```
public void and_and ()
 {
   int a = 10 ;
   int b = 5 ;
   if ((a == 22) && (++b == 5))
     System.out.println("case two");

  System.out.println (b);
 }
```

أيضا هنا لن يقوم المترجم بطباعة ("case two") لأن أحد الشرطين لم يتحقق ولكن عندما يطبع b فإنه سيزيد عليها لأنه قام بفحص المدخل الثاني..

لذا سيطبع لنا ٦ ..

س ٢٩: ما فائدة الطريقة toString()؟

هذه الطريقة موجودة في الفئة Object وبالتالي فهي موجودة في جميع الفئات، لأن كل الفئات ترث من الفئة Object.

هذه الطريقة تنتج صورة نصية للكائن على شكل كائن String

هذا تعريف Sun لطريقة toString

public String **toString**()

Returns a **string representation** of the object. In general, the toString method returns **a string that "textually represents" this object**. The result should be a **concise but informative representation** that is easy for a person to read. It is recommended that all subclasses override this method.

لنفترض لدينا الفئة(الكلاس) التالية

```
public class Person {


        String firstName, lastName;

        int age;


        public Person(String firstName, String lastName, int age) {

                this.firstName = firstName;

                this.lastName = lastName;

                this.age = age;

        }

}
```

إذا نفذنا الكود التالي

```
Person p = new Person("Ahmed", "Ali", 40);

System.out.println(p);
```

سنحصل على النتيجة التالية

Person@3e25a5

حصلنا على هذه النتيجة لإننا لم نعمل override لـtoString الموروثة من Object.

لكن إذا غيرنا الكود السابق إلى التالي

```
public class Person {


	String firstName, lastName;

	int age;


	public Person(String firstName, String lastName, int age) {

		this.firstName = firstName;

		this.lastName = lastName;

		this.age = age;

	}


	@Override

	public String toString() {

		return firstName + " " + lastName + " is " + age + " years 
old.";

	}

}
```

سنحصل على النتيجة التالية

Ahmed Ali is 40 years old.

س ٣٠: كيف أعمل Compile لملف java باستعمال سطر الأوامر؟

نكتب الأمر javac متبوعا باسم الملف

مثلا

javac arabteam.java

س ٣١: كيف أشغل برنامج جافا بعد عملية compile باستعمال سطر الأوامر؟

بعد عملية الـ compile ينتج ملف يحمل نفس اسم ملف .java و لكن ذو امتداد .class

لتشغيل هذا الملف نكتب الأمر java متبوعا باسم الملف (من دون النقطة و الامتداد)

مثال:

لدينا ملف arabteam.class

نكتب

java arabteam

س ٣٢: ما هو مصطلح الـ Casting ؟

هو يعني التحويل من نوع لنوع آخر وذلك بكتابة النوع المطلوب التحويل اليه بين قوسين () امام الاكسبريشين المراد تحويله ، مثل :

```
int a= 35 , b =24 , c =12;
char c;
c = (char) (a+ b+c);
```

س ٣٣: ما هي الطريقة لتعريف الثوابت (Constants) ؟

يستخدم مع الثوابت كلمة final للدلالة على انها قيمة نهائية غير قابلة للتغيير .

مثل : ;final int s = 2

س ٣٤ : ما هو الفرق بين هذه المتغيرات في الجافا ؟

global

local

static

dynamic

global معناها انه أي أحد يقدر يشوف هذا المتغير ..

أما local [يصبح على مستوى البلوك او الفانكشن اللي متعرف في داخلها الفاريبل او على مستوى الكلاس الذي في داخله الفرايبل ..

اما ال Static فهذا متغير يبقى ثابت معاك على طول بنفس القيمة حتى الانتهاء من البرنامج

dynamic هذا متغير تتغير قيمته اثناء ال run -time للبرنامج ..

س٣٥ : ما هو الفرق بين String and Stringbuffer ؟

String:يعتبر كنوع لتعريف متغير نصي ولا يحتاج إلى new ولا يمكن التعديل عليه اثناء تشغيل البرنامج RunTime .

Stringbuffer : لتعريف متغير نصي ايضا ولكنه يحتاج إلى new ويمكن التعديل عليه في أثناء RunTime .

س٣٦ : ما معنى هذا الكلاس في الخط الاحمر ؟

import java.util.GregorianCalendar

هو كلاس محجوز في لغة جافا ويوجد في داخلها عده دوال للوقت والتاريخ وتغير الوقت حسب الموقع .

الوراثة (Inheritance)

س٣٧ : ما معنى الوراثة (inheritance) ؟؟

هي بمفهومها الشائع ان يرث الابن من ابيه بعض او كل صفاته و خاصياته ، فلغة جافا توفر لك هذه الخاصية من خلال امكانية وراثة كلاس من كلاس اخر حيث يصبح

الكلاس الجديد يمتلك كل خواص الكلاس الاول (متغيرات ، دوال ...الخ) ويسمى الكلاس الجديد بـ ساب كلاس (SubClass).

س٣٨ : هل نستطيع الوصول الى كل متغيرات الكلاس الاصلي من الكلاس الساب كلاس (SubClass) عن طريق اسم المتغير مباشرة

اذا كانت برايفت ؟؟

الجواب : لا، لا يمكنك الوصول الى المتغيرات بصيغة برايفت عن طريق اسم المتغير مباشرة بل عن طريق السيت (set) و الجيت (get) الخاصة بالمتغير.

ملاحظة : من المستحسن جعل المتغير بصيغة بروتكتد في الكلاس الاصلي ليسهل عليك الوصول اليه من خلال الكلاسات الوارثة من هذا الكلاس .

الإستثناءات (Exceptions)

س ٣٩: ما هي الإستثناءات ؟

هو حدث يظهر اثناء تنفيذ البرنامج ويعرقل خط سير البرنامج الطبيعي ويتم حمل هذه الاستثناءات للتعامل مع الاخطاء المتوقعة والغير متوقعة .

امثلة على بعض الاخطاء المتوقعة :

برنامج يحاول قراءة لينك غير موجود

برنامج مصمم لقراءة ارقام انتجر من ملف فيجدها من نوع سترينج .

س ٤٠: ما هي عملية حمل الاستثناءات (Exception handle) ؟

١- عندما يجد السستم خطأ يتم إيقاف تدفق تنفيذ البيانات في البرنامج وتسمى بعملية قذف الاستثناء throw the exception .

٢- يتم خلق اوبجكت من نوع خاص يسمي exception ويحمل معلومات عن الخطأ الذي حدث .

٣- بعد ذلك يتم التعامل مع الخطأ والتقاطه وايجاد حل مناسب وتسمى بعملية

catch the exception .

س ٤١: ما هي أنواع الإستثناءات ؟

١- Errors class and its subclasses

للاخطاء الداخلية في البرنامج : مثل انتهاء الذاكرة

٢- Exception class and its subclasses

اخطاء محتملة ومتوقع حدوثها وتسمى Checked exceptions ولابد من التقاطها والا تسبب خطأ في البرنامج

٣-Runtime exceptions and its subclasses

أخطاء يسمح المترجم compiler بتجاهلها وتسمى Uncheck Exception

*- بالنسبة لـ Checked exceptions يتم التقاط الخطأ بطريقتين :

١- Try.. catch blocks

وطريقته :

try الكود المشكوك بحدوث خطأ فيه

catch الكود الذي يجب ان ينفذ إذا حصل وحدث الاستثناء

يتم استخدام finally block دائما يتم تنفيذه بعد الtry بلوك ويستخدم لوضع كود نظيف خالي من الاستثناءات.

٢- تعريف الاستثناء exception في راس الميثود in the method header :

يتم تعريف الاستثناء في راس الميثود باستخدام كلمة Throws يليها نوع الاستثناء ..

public Boolean check format (String fileName) throws EO Exception

---

عام

س ٤٢ : System.out.print ('1'+1);

لماذا الآوت بوت طبع ٥٠ ؟!

لان 1 character

قيمته ب الاسكي كود ٤٩

زي مثلا حرف 'a'

قيمته ٩٧

فلما تسوي

'a' +1'

يطبع لك ٩٨

وهذا جدول آسكي يوجد فيه كل char وقيمته ب الـ Decemal
http://www.unfiction.com/dev/tutorial/ascii.html

س ٤٣: ما هو معنى مصطلح الـ Deprecated ؟

لنفترض أنك كتبت مكتبة، والنسخة الأولى المكتبة تحتوي على الفئة Class1.

في النسخة الثانية من المكتبة، وجدت أن الفئة Class1 ينقصها الكثير من الميزات، فقررت أن تكتب فئة جديدة Class2 التي تقوم بنفس عمل Class1 بالإضافة لأشياء أخرى.

طبعا لن تقوم بمسح Class1 مباشرة، بل أولا ستقوم بوصفها بأنها deprecated، أي أنها متقادمة(قديمة أو مهمله)، حتى يعرف المستعملون أن هذه الفئة متقادمة ويجب استعمال Class2. بعد ذلك، في النسخ القادمة، يمكنك حذفها.

في أكثر الأحيان تستخدم الـ Deprecated لوصف الطرائق Methods المتقادمة والتي ينصح بعدم استخدامها

مثال: في JFrame الطريقة Show هي Deprecated حيث تم استبدالها بالطريقة setVisible

س ٤٤: ما هو الـ Generic class ؟

هو كلاس يتعامل بشكل عام ويحتوي على الأقل على عنصر واحد غير محدد النوع ، يتم تحديده في أثناء إنشاء الأوبجكت .

وهو باختصار القدرة على أني ارسل أو أبعث Type كبراميتر ، وال Generic اقدر استخدمه مع الكلاس أو الميثود ، يعني كأني اقوله أن الميثود اللي استخدمت معها ال Generic أنها راح تاخذ Type غير محدده في وقت إنشائي للكلاس أو الميثود ، طيب يعني متى تحدد له القيمة ؟ راح تحدده له أنت أو المستخدم وهو يستخدم الكلاس أو الميثود ، فعشان كذا هذا يعتبر أمر لل Compiler أنه يشيّك على Compiler Time على القيمة أو ال Type اللي مرسله له ،وطبعا هو قسم كبير جدا بس هذا شرح بسيط له .

س ٤٥: ما هو الـ Interface ؟

هو عبارة عن شيء شبيه بالكلاس العادي إلا انه لايحتوي سوى على متغيرات ثابتة و ميثودات من نوع Abstract أي ميثود هيدر فقط .

* الانترفيس : كي يتم استخدامه لابد ان نعمل implements لكلاس آخر .

* لايمكن خلق اوبجكت من الانترفيس باستخدام كلمة new .

* يستطيع الانترفيس ان يرث من اكثر من انترفيس باستخدام كلمة extends .

* الكلاس العادي غير مسموح له بالوراثة من اكثر من سوبر كلاس ، لكنه يمكن له implements اكثر من انترفيس .

GUI

س ٤٦ : ما هي GUI (واجهة المستخدم الرسومية) في الجافا؟

• عنصر الـ GUI هو كائن يتفاعل معها المستخدم عن طريق ماوس أو لوحة المفاتيح، أو أي نوع آخر من المدخلات.

- ومن الأمثلة الفعلية إذا تعاملنا مع كلاس JOptionPane مثلاً ، نستخدم الميثود showInputDialog() للحصول على مدخلات من المستخدم.

س ٤٧ : ما هو عمل المكون(العنصر) JFrame؟

- JFrame مكون يوفر البنية التحتية لواجهة المستخدم الرسومية (GUI) .
- يوفر JFrame شكل أو سلوك النافذة، فهو يحتوي على شريط العنوان، و أزرار لإغلاق، والتكبير، والتقليل إلى أدنى حد من النافذة.

س ٤٨ : ما هي طرق استخدام الـ JFrame ؟

- إنشاء object (كائن) من نوع JFrame.
- الوراثة (extends) من كلاس (فئة) Jframe .

س ٤٩ : كيفية عمل واجهات مرئية بالجافا باستخدام الكود ؟

- ❖ كيفية عمل Frame عن طريق الكود في الدالة الرئيسية :
- ❖ ملاحظة هذه الكائنات موجودة في مكتبة swing

```
import javax.swing.*;
```

والـ * لإدراج كل الكائنات الموجودة في المكتبة .

```
JFrame Frame2 = new JFrame();

Frame2.setLayout(new FlowLayout());
Frame2.setTitle("عنوان الفريم");

Frame2.setSize(300,300);    // حجم الفريم
```

```
    Frame2.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        Frame2.pack();
        Frame2.setVisible(true);        // إظهار الفريم
        Frame2.setLocation(200,60);    // لتحديد مكان ظهور
النافذة
        Frame2.setResizable(false);   // لمنع تغيير حجم
       النافذة
        Frame2.setBackground(Color.BLUE);        vcvcvcvcvv


    }
```

❖ أو عن طريق الوارثة :

```
public class MainClass extends JFrame {
  public static void main(String[] args) {
    new MainClass().setVisible(true);
  }
```

❖ لعمل خلفية لون للنافذة :

```
    Container C = Arabic.getContentPane();
      C.setBackground(Color.red);
```

❖ كيفية جعل النافذة في منتصف الشاشة : أولا لابد معرفة حجم الشاشة

```
   // import the libraries
   import java.awt.Dimension;
   import java.awt.Toolkit;

   // Get the size of the screen
   Dimension dim =
  Toolkit.getDefaultToolkit().getScreenSize();

// Determine the new location of the window
         int w = Frame2.getSize().width;
         int h = Frame2.getSize().height;
         int x = (dim.width-w)/2;
         int y = (dim.height-h)/2;

         // Move the window
```

```
Frame2.setLocation(x, y);
```

❖ عمل الدالة () Pack تقوم بـ تحجيم الفريم تلقائيا بناء على حجم المكونات الموجودة على الفريم بدلا من استخدام الدالة :

```
Frame2.setSize(width, height);
```

لأنه إذا لم نستخدم هذه الدالة أو دالة setSize فقط سوف يظهر عنوان الفريم كما في الصورة :



Example :

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class Pack extends JFrame {
  private Button button1 = new Button("Button 1");
  private Button button2 = new Button("Button 2");
  private Button button3 = new Button("Button 3");
  private Button button4 = new Button("Button 4");
  private Button button5 = new Button("Button 5");

  public Pack() {
  super("pack() vs. setSize() method Example");
  this.addWindowListener(new WindowAdapter() {
  public void windowClosing(WindowEvent e) {
  System.exit(0);
  }
  });

  Container contentPane = this.getContentPane();
  contentPane.setLayout(new FlowLayout());
  contentPane.add(button1);
  contentPane.add(button2);
```

```
    contentPane.add(button3);
    contentPane.add(button4);
    contentPane.add(button5);

    //this.setSize(450, 90);
    this.pack();
    }

    public static void main(String[] args) {
    Pack mainFrame = new Pack();
    mainFrame.setVisible(true);
    }
}
```



❖ إضافة عنوان JLabel للفريم

```
JLabel t1 = new JLabel("Any text here");
JLabel t2 = new JLabel(message);// message is a String variable
```

- يمكن تغيير العنوان الظاهر على الـ Label باستخدام الدالة setText() method

```
t1.setText("New Text"); // You can also use variables here
```

- الحصول على النص الموجود في العنوان Label باستخدام الدالة :

```
String str = t1.getText();
```

❖ إضافة زر إلى الـ Frame نتعامل مع الـ Button كـ Object واستدعاء دالة البناء مع إرسال لها العنوان الذي سوف يظهر على الزر

```
JButton button = new JButton("Title of Button");
```

o أو إرسال العنوان في الدالة setText(String)

```
button.setText("Title of Button ");
```

o الحصول على النص الموجود في العنوان JButton باستخدام الدالة :

```
String txt = button.getText();
```

o دالة إضافة تلميح للزر من خلال الدالة:

```
b.setToolTipText("click me");
```

o ثم يتم إضافته هذا الكائن إلى الفريم قبل ما يتم عرض الفريم من خلال الدالة setVisible()

```
Frame2.add(b);
```

❖ إضافة حقل نصي JTextField يسمح للتحرير من سطر واحد من النص

```
JTextField tf = new JTextField();  // The default constructor

JTextField tf2 = new JTextField(25); // Constructs the JTextField of size

JTextField tf3 = new JTextField("Hello World", 25); // With size and default text
```

o يمكنك تعيين نص أو استرجاعها من خلال الدالة setText ()

```
tf.setText("New Text"); /* or */ tf.setText(txt);
```

o ويمكن إضافته إلى JFrame

```
frame.add(new JTextField("Hello World", 25));
// or //
frame.add(tf);
```

o للحصول على النص الموجود في الحقل النصي من خلال الدالة getText ()

```
String Stext = JTextField1.getText();
```

- إضافة قائمة combbox للفريم : عن طريق مصفوفة عناصر

```
JComboBox com = new JComboBox();
  s = new String []{"sasa","sasfg","dd","aa"};
  com.setModel(new javax.swing.DefaultComboBoxModel(s));
  Frame2.add(com);
```

- إضافة عنصر للقائمة من حقل نصي JTextField من خلال الدالة :JComboBox.addItem(Object item)

```
      tf3 = new JTextField("Hello World", 25); // With
  size and default text

      Frame2.add(tf3);
      com = new JComboBox<Object>();
      Frame2.add(com);
      com.addItem(tf3.getText()); // add to CombBox from
JTextField
```

- إضافة عنصر للقائمة من متغير نصي : لان متغير من نوع String يعتبر Object لان كل ما في الجافا هي عبارة عن Objects

```
      String comtxt="Add in Combobox";
      com.addItem(comtxt);
```

- لجعل القائمة CombBox قابلة للتعديل من خلال الدالة setEditable

```
      com.setEditable(true);
```

- إضافة قائمة list للفريم

```
  list = new JList(data); //data has type Object[]
```

- مع إضافة قيم لها من مصفوفة

```
String[] ar = {"one", "two", "three"};
JList list = new JList(ar);
Frame2.add(list);
```

- يمكن إضافة عناصر للـ list من خلال الكلاس ListModel class لابد اولا تعريف الـ DefaultListModel

```
DefaultListModel listModel = new DefaultListModel();
listModel.addElement("item 1");
listModel.addElement("item 2");
```

- ويمكن استخدام الـ DefaultListModel في دالة البناء للـ list

```
JList list = new JList(listModel);
```

- الآن لإضافة عناصر إضافية في أي وقت أثناء تنفيذ باستخدام *addElement() method*

```
listModel.addElement("new item");
```

- يمكن إضافة عنصر في القائمة بتحديد موقعها

```
int index=3;
listModel.add(index, object);
```

- الـ object سواء متغير نصي أو قيمة من حقل نصي أو غيرة
- لحذف عنصر معين من الـ list حيث الـ index عنوان العنصر المراد حذفه

```
listModel.remove(index);
```

- لحذف كل العناصر:

```
listModel.clear();
```

- العنصر المحدد في الـ list

```
list.setSelectedItem(n);
```

- لإرجاع سلسلة المحدد حاليا في JList، استخدم

```
s = (String) list.getSelectedValue();
```

ويعرف هذا الأسلوب يعيد كائن object لأن JList في الواقع يمكن أن تحتوي على أنواع الكائنات الأخرى من سلاسل (على سبيل المثال، الصور الصغيرة تعرف باسم الرموز

- للتحديد المتعدد في الـ JList

```
JList.setSelectionMode(ListSelectionModel.MULTIPLE_INTERVAL_SELECTION);
```

حيث الخاصية ListSelectionModel لها أكثر من قيمة :

```
- MULTIPLE_INTERVAL_SELECTION //(default select)
```
تحديد متعدد للعناصر

```
- SINGLE_INTERVAL_SELECTION    //
```
تحديد واحد فقط

```
- SINGLE_SELECTION
```

```
// Create a list that allows adds and removes
DefaultListModel model = new DefaultListModel();
JList list = new JList(model);

// Initialize the list with items
String[] items = {"A", "B", "C", "D"};
for (int i=0; i<items.length; i++) {
    listModel.add(i, items[i]);
}

// Append an item
int pos = list.getModel().getSize();
listModel.add(pos, "E");

// Insert an item at the beginning
pos = 0;
listModel.add(pos, "a");
```

- لتغير محتوى الموقع المحدد

```
pos = 1;    // index of item to replace it
listModel.set(pos, "b");
```

- لحذف العناصر محدد حسب الـ index

```
// Remove the first item
pos = 0;    // any index of item
listModel.remove(pos);
```

- لحذف أخر عنصر في القائمة

```
// Remove the last item
pos = listModel.getSize()-1;
if (pos >= 0) {
    listModel.remove(pos);
}
```

- لحذف كل العناصر في الـ list:

```
// Remove all items
listModel.clear();
```

س ٥٠ : ما هي الـ Swing؟؟

هي مكتبة لأدوات الواجهة الرسومية (مثل : الأزرار ، مربعات النص ...) تختصر إلى GUI و هي من إنتاج Sun Microsystems ، تكون مضمنه مع الـ Java باسم Swing .

تتميز بأنها:

متعددة المنصات ، أي تعمل على أي بيئة تشغيل.

قابله للتعديل

قابليه للتجديد

خفيفة

س ٥١ :ما هي مكتبة SWT ؟

هي مكتبه للـ GUI في لغة الـ Java ، وتسمى SWT . تم برمجتها في شركة IBM و الأن هي مشروع مفتوح المصدر ، مدعمه من IBM ، تعتبر الـ SWT مثال للأدوات الثقيلة (Heavyweight) ، وتسمح باستغلال أساس نظام التشغيل لعمل وجهه رسوميه GUI ، وتقوم بإنشائها باستخدام الواجهة الأصلية للـ Java ، من مُمَيزاتها السرعة و التأثيرات الجميلة التي يُطلق عليها أسم الـ Look and Feel ، و من سلبياتها كثرة احتمالات الأخطاء بها و أقل كفاءه من الـ Swing ، و الـ SWT تماماً مثل مكتبة Windows Centric .

س ٥٢ : لماذا دائما يفضل أن تكون الـ method داخل الـ class تكون static ؟

لكي استطيع ان استخدمها خارج الكلاس من غير ان انشأ أوبجكت من الكلاس فقط بواسطة اسم الكلاس واسم الفنكشن مثلا className.method .

س ٥٣ : كم مكتبة للواجهات لدى الجافا ؟

1. swing المكتبة المعتمد حاليا لدى الجافا

http://docs.oracle.com/javase/tutorial/uiswing/ swing

2. awt هي مكتبة قديمة انتهى دعمها من قبل الجافا

http://docs.oracle.com/javase/6/docs/technotes/guides/awt/

3. swt المكتبة التي انتجتها IBM

http://www.eclipse.org/swt/

4. gwt المكتبة الخا صة بـgoogl لبناء تطبيقات الويب

http://www.gwtproject.org/overview.html

5. javafx UI تقنية أخرى أصدرتها صن مايكروسيستمز لتطوير تطبيقات من نوع RIA .

http://docs.oracle.com/javafx/

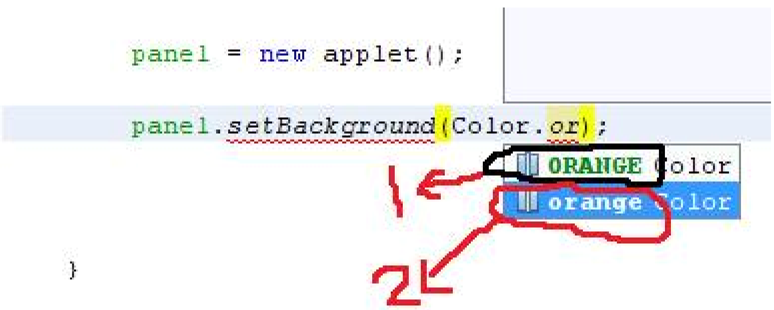
6. QT وتسمي qt-jambi

http://qt-jambi.org/

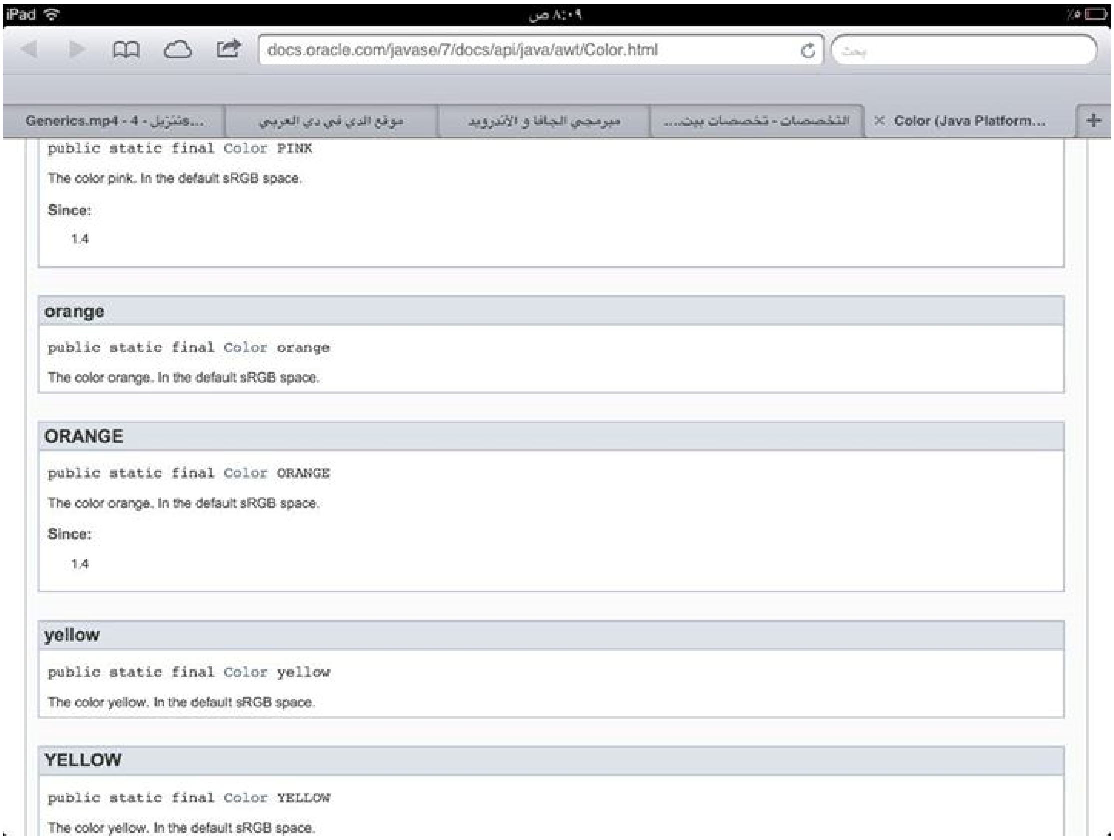
7.GTK

http://java-gnome.sourceforge.net/

س٥٤ : ما هو الفرق ما بين orange 1 , orange 2 ؟



؟؟؟



لو تلاحظون في الصورة أن داخل كلا الثابتين في كلاس Color ما فيه بينهم فرق إلا في حالة الأحرف ، وكان في السابق لا يوجد إلا الثابت orange(lowercase) ولكن بعد JDK 1.4 تمت اضافة ORANGE(uppercase) ، لكي ينفذوا مصطلح أن الثوابت دائما تكون Upper_Case (اتفاقية ترميز الثوابت).

س٥٥ : كيف يمكن المناداة على المتصفح الإفتراضي و فتح صفحة ويب معينة؟

الجافا ومنذ نسختها السادسة، تتيح إمكانية المناداة على المتصفح الإفتراضي، وذلك عن طريق الفئة (class) Desktop التي توفر لنا الـ Method (الطريقة) browse

```
//get a Desktop object

Desktop desktop = Desktop.getDesktop();
```

```
try {

        //open the default browser using the method browse which take an URI
object representing the web page

        desktop.browse(new URI("http://arabteam2000-forum.com/"));

} catch (URISyntaxException ex) {

        ex.printStackTrace();

}
```

س ٥٦: كيف أجعل واجهة برنامجي متعددة اللغات؟

ما نريده هو إظهار كل رسائل ونصوص الواجهة حسب اللغة التي يتم اختيارها. إذن لو كنا سنجعل ثلاث لغات متوفرة في البرنامج، فسيلزمنا ثلاث نسخ من كل نص ورسالة، كل نسخة بلغة مختلفة. سنضع كل نسخة من النصوص في ملف properties مستقل، ما يعني أن البرنامج لن يحتوي على أي نص، بل كل شيء سيكون في الملفات properties.

س ٥٧: ما هي الملفات properties ؟

عند كتابة هذه الملفات، يجب احترام نموذج محدد: أولا إسم الملف، مثلا "labels"، متبوعا بالرمز "_"، متبوعا بالرمز الخاص باللغة، مثلا رمز العربية هو "ar"، رمز الفرنسية "fr" ورمز الإنجليزية هو "en". إذن حسب النموذج فإن إسم ملف اللغة العربية يجب أن يكون كالتالي labels_ar.properties

محتوى هذه الملفات يكون على شكل

key=value

key=value

....

بحيث أننا نسترجع النص الذي نريده بواسطة key.

كما أنه تجدر الإشارة إلى أن الملفات properties تقبل فقط الترميز ISO-8859-1، أي الحروف اللاتينية فقط، إذا أردنا إضافة حروف عربية، يجب أن نكتب الرمز unicode المرادف لكل حرف.

س ٥٨: ما هي فئة الـ java.util.Locale ؟

هذه الفئة تمكننا من تحديد المنطقة الجغرافية أو الثقافية التي نريد. فبما أننا نريد أن نغير لغة واجهة برنامجنا، فيلزمنا وسيلة لتحديد هذه اللغة، وهنا تظهر فائدة الفئة Locale. مثلا إذا أردنا إنشاء Locale خاص بالعربية، فإننا نمرر رمز اللغة العربية للconstructor .

```
Locale arLocale = new Locale("ar");
```

لمعرفة الرمز الخاص بكل لغة http://www.loc.gov/standards/iso639-2/php/code_list.php <--

س ٥٩: ما هي فئة الـ java.util.ResourceBundle ؟

هذه الفئة هي التي تقوم بتحميل ملف الproperties الخاص باللغة التي نحددها، وبالطبع تحديد اللغة يعني إنشاء كائن Locale خاص بهاته اللغة. بعد تحميل الملف فإننا نستطيع الوصول للقيم الموجودة بداخله عن طريق getString التي نمرر لها قيمة الkey.

للحصول على كائن ResourceBundle فإننا ننادي على الطريقة getBundle، ونمرر لها إسم الملف properties، لكن ليس كل الإسم، لكن فقط الجزء الموجود قبل اللغة، مثلا إذا كان لدينا الملف labels_en.properties، فإننا نمرر "labels" فقط للطريقة getBundle. هذه الطريقة تنتظر معطى آخر، وهو كائن Locale يمثل اللغة التي نريد استعمالها.

```java
Locale currentLocale = new Locale("ar");

ResourceBundle myResources = ResourceBundle.getBundle("labels", currentLocale);

String someValue = myResources.getString("someKey");
```

Java I/O

س ٦٠: كيف أقرأ محتوى ملف نصي سطرا بسطر؟

لقراءة ملف نصي سطرا بسطر، يمكننا استعمال الفئة BufferedReader أو الفئة Scanner:

باستعمال الفئة BufferedReader والطريقة readLine:

أولا ننشئ كائن BufferedReader ونمرر لـconstructor كائن من فئة FileReader الذي نمرر له بدوره مسار الملف الذي نريد قراءته

```java
BufferedReader reader = null;

//...

reader = new BufferedReader(new FileReader("myfile.txt"));
```

ثم نقوم داخل loop بقراءة سطر من الملف وتخزينه في متغير من فئة String، ونقوم بعد ذلك بطباعة قيمة هذا المتغير.

الloop تتوقف عندما تعيد الطريقة ()readLine القيمة null، مما يعني أننا وصلنا إلى نهاية الملف.

```
String line;
while ((line = reader.readLine()) != null) {
        System.out.println(line);
}
```

الكود كاملا:

```
BufferedReader reader = null;
try {
        reader = new BufferedReader(new FileReader("myfile.txt"));
        String line;
        while ((line = reader.readLine()) != null) {
                System.out.println(line);
        }
} catch (IOException ex) {
        ex.printStackTrace();
} finally {
        try {
                if (reader != null) {
                        reader.close();
```

```
                }
        } catch (IOException ex) {
                System.out.println("Closing stream failed.");
        }
}
```

س ٦١: ماذا وضعنا أمر إقفال reader داخل finally؟ ماذا سيحصل لو وضعنا أمر الإغلاق داخل ال try block، بعد الإنتهاء من الloop، كما الكود التالي؟

```
//...
String line;
while ((line = reader.readLine()) != null) {
        System.out.println(line);
}
reader.close()
//...
```

لنفترض مثلا أن البرنامج عند تنفيذ الأمر readLine واجه مشكلة في قراءة الملف، مما جعله يرفع IOException، هذا يعني أنه سيتجاهل كل الكود الموجود بعد السطر الذي رفع الإستثناء وسينفذ مباشرة الكود الموجود في catch bloc، وبالتالي لن ينفذ أمر الإغلاق، وسيبقى الreader مفتوحا، وبالتالي لن يتم تحرير الذاكرة التي يستعملها.

الحل إذن هو وضع إمر الإغلاق داخل finally bloc، لأن الكود الموجود داخله يتم تنفيذه دائما، سواء تم رفع exception أو لا.

س ٦٢: كيف نقرأ الملف باستعمال الفئة Scanner ؟

أولا ننشئ كائن Scanner ونمرر له الملف الذي سنقرأه

```
Scanner scanner = new Scanner(new File("myfile.txt"));
```

ثم نقوم داخل loop بقراءة الملف سطرا سطرا بواسطة الطريقة nexLine()، لكن أولا يجب أن نتحقق أننا لم نصل إلى نهاية الملف .

```
while (scanner.hasNextLine()) {
        String line = scanner.nextLine();
        System.out.println(line);
}
```

ثم نقفل الscanner

```
scanner.close();
```

الكود كاملا :

```
Scanner scanner = new Scanner(new File("myfile.txt"));
while (scanner.hasNextLine()) {
        String line = scanner.nextLine();
        System.out.println(line);
}
scanner.close();
```

عام

س٦٣ : ما هي الـ JPA ؟

Java Persistence APIs ،

framework -إن صح التعبير- للتعامل مع الداتا بيس (Database) بطريقة OOP . أي أنك لا تتعامل مع الداتا بيس باستعلامات عادية -خاصة في حالة الإضافة والتعديل- ومع ذلك يوجد استعلامات للبحث والحذف..

JPA هي تحقيق لمفهوم ORM (Object Relationship Mapping) أي التعامل مع الداتا بيس بطريقة غرضية .

س ٦٤ : ما هو الفرق بين JDK و JRE و JVM ؟

حاول ان تجاوب على حسب معرفتك ثم قارن بين الاجابتين ، واذا كانت هناك اي اضافة او شرح تفصيلي ، فكلنا نستفيد مع بعض ..

هذا السؤال غالبا المبرمجون ما يهتمون فيه كثيرا ولا في كيفية عملية ترجمة البرنامج !

|
|
|
|
|
|

JDK (Java Development Kit)

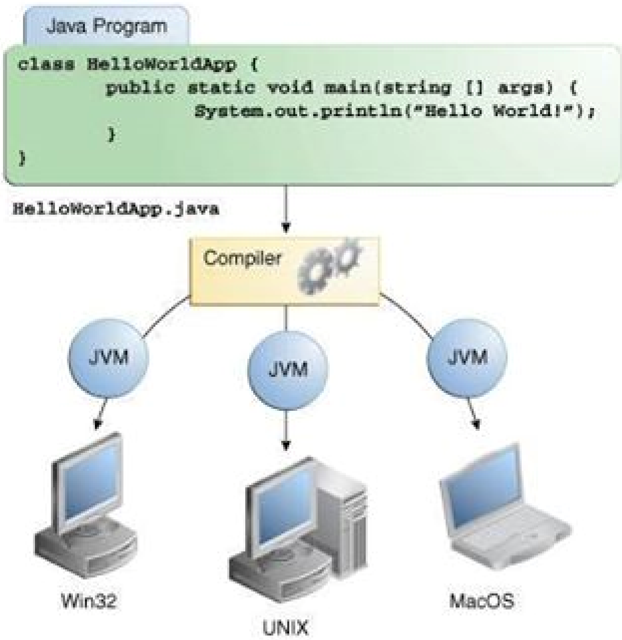
يندرج تحته الادوات الخاصة بعمل برامج باستخدام لغة الجافا ،
وهذه الادوات تحتوي أيضا على Compiler الخاص بلغة الجافا
ولتشغيل هذه البرامج على الجهاز المستخدم لابد من وجود jre

JRE (Java Runtime Environment)

تحتوي هذه الباكدج على المكتبات والملفات الخاصة بلغة الجافا وايضا الملفات المدعومة سواء افتراضيا مع الجافا او اضيفت عن طريق المبرمج ، ولكنها لا تحتوي على أي أدوات تستخدم في عملية التكويد مثل الكومبيلر و غير ذلك . ولهذا اذا اردت تشغيل اي برنامج للجافا لا بد ان يكون على جهازك الخاص هذه الباكدج

JVM (Java Virtual Machine)

بعد ما حول ملف java. من إلى class.
وهذا الملف مكون من بايت كود طبعا نظام التشغيل لا يفهمه
فتقوم jvm بتحويل ملف class. من Byte code إلى Binary (لغة الاله)



Data Structure

س ٦٥: أول سؤال ممكن يكون على بالك أن هل الـ Data Structure لها علاقة بالـ Database ام لا ؟

الإجابة لا ، هناك فرق بين ما يعرف بقواعد البيانات Databases ، وبين مفهوم هياكل(تراكيب)البيانات Data Structure .

قواعد البيانات هي عملية تخزين البيانات .

أما هياكل البيانات هي اللي تحتاجها لأداء برنامج بكفاءة عالية في وقت التنفيذ (خلال الـ runtime).

بمعنى أوضح قواعد البيانات أنت تخزن فيها البيانات لأجل أنك لو أغلقت الكمبيوتر تجي اليوم الثاني تلقاها موجودة .

فيه فرق بين تخزين بيانات وبين بيانات حية في الذاكرة للتعامل معها .

في البرنامج عادة العمليات التي تتم في RunTime تتم على Data Structure.

هياكل البيانات باختصار هي تحليل شكل البيانات شلون تكون داخل الاوبجكت ونستخدمها لجمع البيانات وترتيبها وتنظيمها ...

في جافا فيه اوبجكت جاهزة لك أنك تستخدمها على طول يعني ما يحتاج تعيد كتابة كل شئ من الصفر مثل ما هو في C و ++C ، كذلك نتكلم عن أن هياكل البيانات يأتي معها خوارزميات للترتيب وإعادة الترتيب وصياغة أجزاء البيانات الموجودة ، وهذه الخوارزميات موجودة مع الاوبجكت يعني مضغوطة مع الاوبجكت او الانترفيس اللي راح نتكلم عنه الآن .

مركز العمليات هو انترفيس interface رهيب جداً هو الـ Collection .

الشكل الأساسي للـ Collection interface هو أنه يتفرع منه اثنين من الـ interface : وهي Set و List .

Set معناها مجموعة من الأشياء أو البيانات لا يوجد فيها تكرار .

ما تقبل التكرار اذا كان ConCrete Class لازم تعرف فيه الـ Two Method الموجوده في كل اوبجكت اللي هي equals و hashcode .

List معناها مجموعة من الأشياء والبيانات وتسمح بالتكرار وبمزايا أخرى .

أيضا يوجد الـ Map Interfaca لكنه ليس من Collection فهو جسم برمجي آخر عبارة عن دمج جسمين مع بعض .

نقطة مهمه أن هذه الانترفيس Set و List لا يمكن انك تسوي منها اوبجكت يعني ما تقدر تسوي من عندها new مثلا new Set أو new List ،

لأنها كأي انترفيس لا تستطيع ان تنشأ منه اوبجيكت والدوال اللي فيه تطبقها كلاسات ConCrete يعني اوبجكت جاهزة للاستخدام فتستطيع

إذا تبي :

Set تقدر تستعمل أما HashSet أو TreeSet المطبِّقان له .

List تقدر تستعمل أما ArrayList أو LinkedList .

Map تقدر تستعمل HashMap أو TreeMap .

Tree معناتها Sorted يعني مرتبة حسب ترتيب معين انت حددته سابقاً .

ملاحظة : أني اتكلم الآن عن الشكل الأساسي أما الآن في الـ Collection فيه انترفيس جديد Queue وفي Set و List كلاسات أخرى جاهزة

للاستخدام غير اللي ذكرتهم . يمكنك مشاهدة الصورة في آخر المقالة وتشاهد Collection وتفرعاته .

لو أخذنا الشكل الأساسي بشكل مكبر قليلاً نشوف أن Collection مرتبط في شئ اسمه Iteratable و Iterate يعني انه قابل انه يدور

وايضاً عندي من Iteratable Interface اوبجكت اسمه --> Iterator وهذا الاوبجكت اقدر استدعيه عن طريق الكونستراكتر Constructor .

Iteratable Interface :

يوجد فيه عملية واحدة فقط اللي هي (Iterator() كونستراكتر) ، يوجد في الاوبجكت أو كلاس Iterator ثلاث عمليات :

boolean hasNext()

لأجل تحديد الوجه للوب Loop .

boolean next()

يعطيك الأوبجكت التالي

void remove()

اذا تبغى تشيل (تحذف) كائن .

أول شئ الـ collection interface من الباكج java.util .

.

ندخل الآن في عمليات الـ collection interface :

١_ Informative Methods تحتوي على :

Iterator Iterator()

عرفناها فوق

boolean isEmpty()

هنا نسأل هل الكولكشن collection فاضي ويرجع لنا boolean (true Or false)

int size()

يرجع لنا الحجم .

٢_ Object Based Methods :

boolean equals(Object o)

نستعلمها في المقارنة بين اثنين من الكولكشن .

int hashCode()

يرجع لنا الرقم الخاص بالاوبجكت .

٣_ Element Based Methods :

boolean add(Object o)

إذا ابغى اضيف كائن .

boolean remove(Object o)

إذا ابغى احذف كائن .

boolean contins(Object o)

إذا اشيك هل تحتوي على .

٤_ Output To Arrays Methods :

Object[] toArray()

إذا خلصت من الأوبجكت وابغى احوله إلى Arrays .

Object[] toArray(Object[] o)

نفسها لكنها تحول مصفوفة وترجعها لنا .

٥_ Collection Based Methods :

هذي عمليات خاصة بين كولكشن وكلوكشن مثلا A و B .

boolean addAll(Collection c)

تعطي كولكشن تضاف بالكامل على الكولكشن المراده .

boolean containsAll(Collection c)

مثلا هل A تحتوي على كل عناصر B .

boolean removeAll(Collection c)

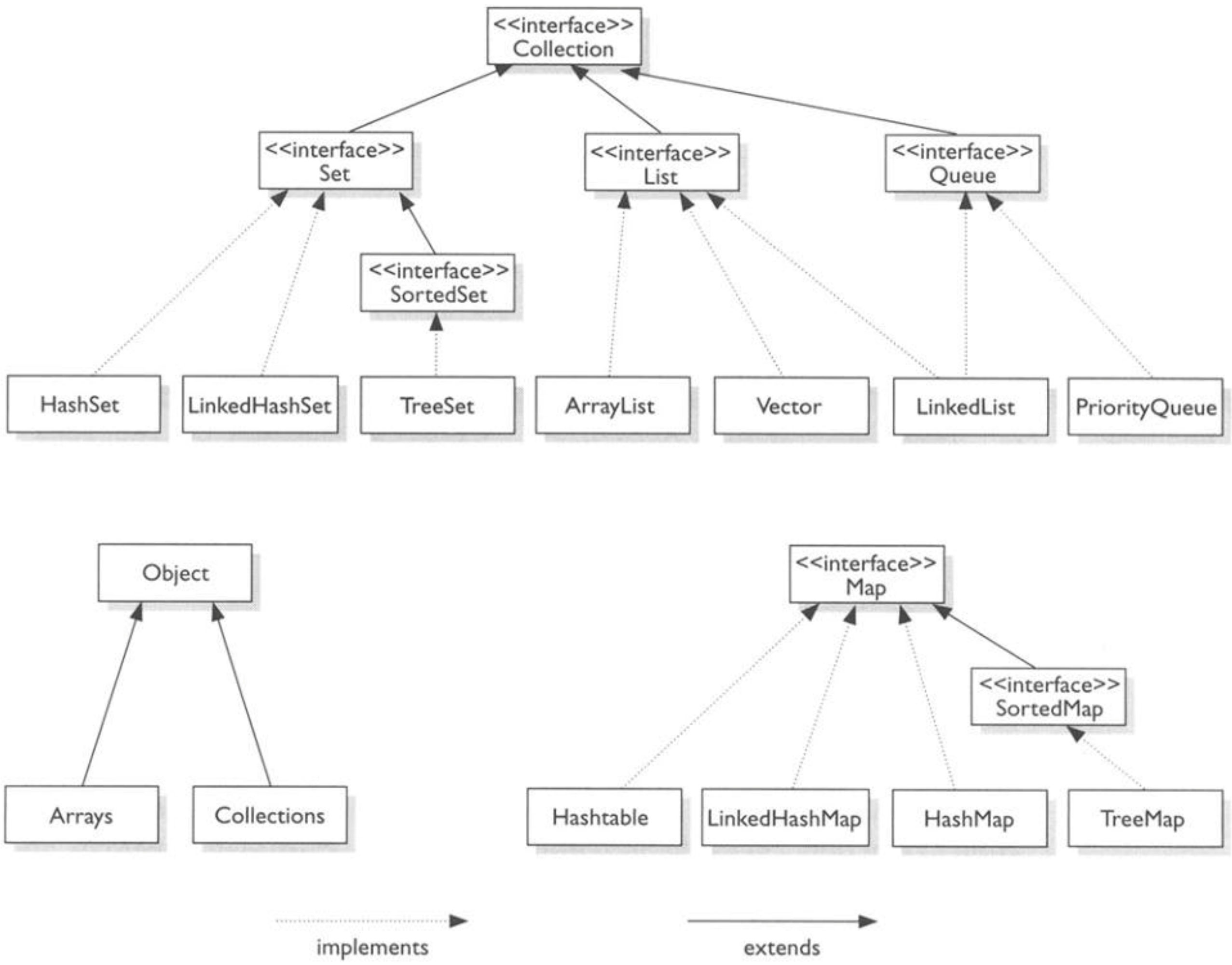
أنا ممكن اقوله A removeAll B فسيحذف كل العناصر غير الموجودة أو المشتركة .

void clear()

راح ينظف A يعني راح يفضيها من الداخل.

boolean retainAll(Collection c)

هنا نقوله حافظ على اللي موجود في A و B .



س ٦٦ : ما هو الفرق بين الـ arrayList و الـ LinkedList ؟

لو تلاحظ الصورة تشوف أن ArrayList و LinkedList من نفس الانترفيس List ، يعني متشابهات جدا في الاستخدام، ولكن LinkedList لديه تفرع من الانترفيس Queue ، وهذا يعطيها مميزات (دوال) أكثر من ArrayList مثل poll , peek , offer وغيرها ، الفرق الرئيسي هو التنفيذ (implementation) اللي يسبب أداء

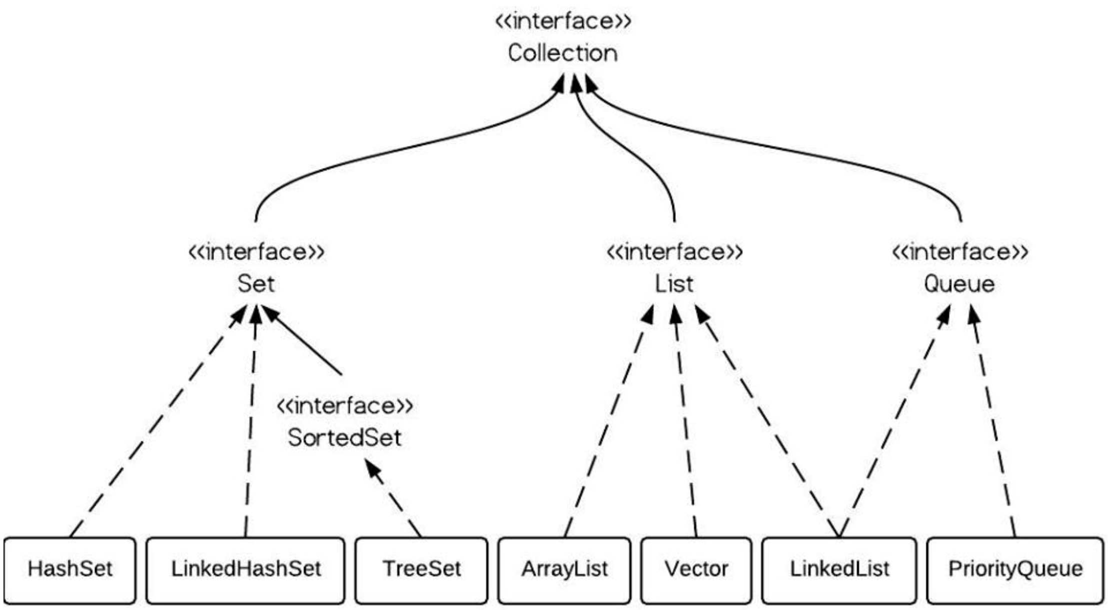
مختلف للعمليات المختلفة ، تستعمل ال ArrayList كarray تتغير حجمها واضافة عناصر جديدة ويزداد حجمها بشكل ديناميكي ، ايضا يمكن الوصول إليها مباشرة عن طريق دوال set و get ، وال LinkedList تستعمل أو تنفذ كقائمة مزدوجة مرتبطة مع بعضها ، أدائها سئ للدوال set و get ولكن أدئها أفضل في الاضافة والإزالة أفضل من ال arrayList .. وهناك اختلافات أخرى ..

وهذا مثال بسيط لاستعمال LinkedList :

```
List<String> staff = new LinkedList<>(); // LinkedList implements List

staff.add("Amy"); staff.add("Bob"); staff.add("Carl");  Iterator iter = staff.iterator();

String first = iter.next(); // visit first element String second = iter.next(); // visit second element iter.remove(); // remove last visited element
```



س ٦٧ : ما هو الفرق بين

List<String> list = new LinkedList<String>();

و

LinkedList list<String> = new LinkedList<String>();

؟

الفرق انك ممكن تحتاج نفس ال LinkedList في الأسفل لكن يكون وقتها تبغى تستخدمها ك ArrayList أو انك تكون تبغى تحصل على القيم مثل ما انا واضع في الشرح فوق ، فبالوقت هذا تكون ArrayList هي الأنسب ونفس الشئ لو كان الوقت انسب اني استبدل ArrayList إلى LinkedList ، فإسناد ال LinkedList أو ArrayList إلى List interface ، لاجل اني لو ابغى ابدل بينهم في وقت لاحق ..

س ٦٨: كيف أقوم بترتيب (sorting) عناصر المصفوفة؟؟

من أجل ترتيب مصفوفة ما، يمكننا استعمال الطريقة Arrays.sort والتي تنتظر كargument المصفوفة التي نريد ترتيبها

مثلا :

```
int[] array1 = {10, 3, 4, 45, 2, 0, 9};

System.out.println("array1 before sorting: " + Arrays.toString(array1));

Arrays.sort(array1);

System.out.println("array1 after sorting: " + Arrays.toString(array1));
```

والنتيجة هي كالتالي

```
array1 before sorting: [10, 3, 4, 45, 2, 0, 9]
 array1 after sorting: [0, 2, 3, 4, 9, 10, 45]
```

ملاحظة :يجب أن تكون عناصر المصفوفة إما primitive types أو فئات تعمل implement للواجهة Comparable وتعرف الطريقة compareTo

---

عام

س ٦٩: ما هو Java Annotations ؟

ال Annotation، أو (حسب معجم ArabEyes) التعليق/الحاشية، هو إضافة معلومات إلى الشيفرة. لكن الفرق بينها وبين ال comments أو ال Javadoc، هو أن ال annotations يمكن استعمالها من طرف annotation processor للقيام بعمليات مختلفة: إنشاء ملفات، إنشاء جداول في قاعدة البيانات، التحقق من احترام الشيفرة للمعايير المحددة، أو حتى إضافة أسطر جديدة من الكود.. وقد تمت إضافة ال annotations إلى الجافا في النسخة الخامسة JavaSE 5.

هناك نوعين من ال annotations، الحواشي الموجودة في الجافا، وهي ثلاثة فقط، والحواشي التي يكتبها المبرمج.

الحواشي (Annotation) الموجودة في الجافا:

@Override: تقوم هذه الحاشية بالإشارة إلى أن الطريقة التي تأتي بعدها تقوم ب override لطريقة في الفئة الأم. إذا لم يجد ال compiler أي طريقة بنفس التوقيع (method signature) في الفئة لأم فسيرسل رسالة خطأ.
مثال: لنفترض الفئة التالية، والتي تعمل override للطريقة toString الموجودة في الفئة Object. سنقوم عمدا بكتابة ال "S" الموجودة في toString ب lower case بدل upper case، ونرى ما النتيجة.

```
▪ public class TestOverride {
▪         @Override
▪         public String tostring() {
▪                 return "Nothing special";
▪         }
```

```
}
```

عند عمل ال compilation، فإن ال compiler يطبع رسالة الخطأ التالية:

TestOverride.java:2: method does not override or implement a method from a supertype

@Override

^

1 error

مع أن استعمال الحاشية @Override اختياري، إلا أنه ينصح باستعمالها دائما، حتى لا نقوم سهوا بإنشاء طريقة جديدة ومستقلة، في الوقت الذي نظن أننا قمنا ب override لطريقة موجودة في الفئة الأم.

@Deprecated: هذه الحاشية تشير إلى أن الطريقة أصبحت deprecated، وهذا لا ينتج عنه خطأ في ال compilation مثل @Override، لكن فقط إنذار (compiler warning).
مثلا، عندما نقوم بال compilation للكود التالي

```
public class TestDeprecated {

        public TestDeprecated() {
                DeprecatedCode d = new DeprecatedCode();
                d.someMethod();
        }
}

```

```java
class DeprecatedCode {

        @Deprecated
        public void someMethod() {
                System.out.println("I am so deprecated!!");
        }
       }
```

فسنحصل على الرسالة التالية من ال compiler:

.Note: TestDeprecated.java uses or overrides a deprecated API

Note: Recompile with -Xlint:deprecation for details
.

وعند إضافة الخيار -Xlint

javac -Xlint TestDeprecated.java $

TestDeprecated.java:4: warning: [deprecation] someMethod() in DeprecatedCode has been deprecated

;()d.someMethod

^

١ warning

@SuppressWarnings: تقوم هذه الحاشية بإلغاء كل الإنذارات الخاصة بالcompiler. مثلا

```
▪ public class TestSupressWarnings {
▪
▪         public TestSupressWarnings() {
▪                 java.util.List<String> list = new java.util.ArrayList();
▪         }
▪        }
```

هذا الكود يعطينا إنذارا من نوع unchecked نتيجة هذا السطر

```
java.util.List<String> list = new java.util.ArrayList();
```

إذا كنا متأكدين مما نعمل وأردنا إزالة الإنذار، يكفي إضافة ،
@SuppressWarnings
وتحديد نوع الإنذار الذي نريد إزالته، وستتم عملية الcompilation
بدون أي إنذار:

```
public class TestSupressWarnings {


        @SuppressWarnings("unchecked")
        public TestSupressWarnings() {
                java.util.List<String> list = new java.util.ArrayList();
        }
}
```

القوة الحقيقة لل Annotations ليست في تلك المتاحة من طرف الجافا، لكن في تلك التي يمكن للمبرمج كتابتها..

س ٧٠: كيفية برمجة حواشي جديدة ؟

طريقة صياغة الحاشية تكون عادة كالتالي

```
//meta-annotations
public @interface AnnotationName {
        //Annotation attributes
}
```

حيث AnnotationName هو إسم الذي نريد إعطاءه للحاشية.

ال meta-annotations

ال meta-annotations هي حواشي تعطي معلومات عن الحاشية التي نحن بصدد كتابتها (annotating the annotation).

الفرق بينها وبين الحواشي المعيارية التي تكلمنا عنها سابقا (Override, @Deprecated, @@SuppressWarnings) هو أن الحواشي المعيارية تصف الكود، بينما ال meta-annotations تصف الحواشي فقط، ولا تستعمل لوصف الكود.

يوجد أربع meta-annotations، وهي كالتالي

@Target, @Retention, @Inherit, @Documented

Target@: تحدد مكان تواجد الحاشية، ويمكن أن نمرر لها القيم التالية:

ElementType.CONSTRUCTOR: يمكن تطبيقها على الconstructor.

ElementType.FIELD: يمكن تطبيقها على الحقول.

ElementType.LOCAL_VARIABLE: يمكن تطبيقها على المتغيرات المحلية.

ElementType.METHOD: يمكن تطبيقها على تعريف الطرق.

ElementType.PACKAGE: يمكن تطبيقها على تعريف الحزمة.

ElementType.PARAMETER: يمكن تطبيقها على المدخلات الخاصة بطريقة أو constructor.

ElementType.TYPE: يمكن تطبيقها على فئة، واجهة، حاشية أو enum.

مثلا لو أردنا كتابة حاشية خاصة بال constructor فقط، سيكون شكلها كالتالي:

```
@Target(ElementType.CONSTRUCTOR)
public @interface ConstructorAnnotation {
}
```

وفي حالة أردنا أن تكون الحاشية خاصة بال constructor والطرق معا:

```
@Target({ElementType.CONSTRUCTOR, ElementType.METHOD})
public @interface ConstructorAnnotation {
}
```

عندما لا نحدد@Target ، فهذا يعني أنه سيتم تطبيق الحاشية على جميع العناصر التي تم ذكرها في الأعلى .

@Retention: هذه ال meta-annotation تقوم بتحديد مدى (scope) الحاشية، وهي تقبل القيم التالية:

RetentionPolicy.SOURCE: لا يتم تخزين الحاشية في الملفات .class، وبالتالي يمكن الوصول إليها فقط من طرف الأدوات التي تستعمل الملف المصدري (compiler, javadoc، ...). مثلا

@Override, @SuppressWarnings.

RetentionPolicy.CLASS: يتم تخزين الحاشية في الملف .class، لكن يتم إلغائها من طرف ال Virtual Machine عند تشغيل البرنامج.

RetentionPolicy.RUNTIME: يتم تخزين الحاشية في الملف .class، ولا يتم إلغائها من طرف ال Virtual Machine عند تشغيل البرنامج، وبالتالي يمكن الوصول إليها باستعمال الreflection.

مثال:

```
@Target(ElementType.CONSTRUCTOR)

@Retention(RetentionPolicy.SOURCE)

public @interface ConstructorAnnotation {}
```

@Inherit: تشير إلى أن هذه الحاشية يمكن أن ترثها الفئات الفرعية (subclasses) من الفئة التي تم تطبيق الحاشية عليها. السلوك الإفتراضي هو عدم التوريث.

@Documented: يقول لل javadoc بإظهار هذه الحاشية في التوثيق الذي سيتم توليده.

خصائص الحاشية:

يمكن أن تكون الحاشية خالية من أي خاصية، وفي هذه الحالة تسمى Markup Annotation. مثلا:

```
public @interface AMarkupAnnotation {

}
```

أو يمكن أن تحتوي على واحدة أو عدة خصائص. هذه الخصائص تكتب على شكل طرق (method) فارغة.

كل خاصية يمكن أن تتوفر على قيمة افتراضية. قيمة هذه الخصائص يجب أن تكون من أحد هذه الأصناف فقط:

الأصناف البدائية (primitive types).

String

java.lang.Class

enum

java.lang.annotation.Annotation

مصفوفة مكونة من أحد الأصناف السابقة.

يتم تعريف خاصية الحاشية كالتالي :

```
public @interface AnAnnotation {
        //خاصية بدون قيمة افتراضية.
        int id();
```

```
        //خاصية تمتلك قيمة افتراضية.
        String description() default "This is an annotation for testing";
}
```

وعند استعمال الحاشية، نقوم بتحديد قيمة الخصائص بالشكل التالي

```
@AnAnnotation(id=1)
public class SomeClass {
        //...
}
```

أو

```
@AnAnnotation(id=1,description="non default description")
public class SomeClass {
        //...
}
```

س ٧١ : كيفية معالجة الحواشي (Annotation Processing) ؟

معالجة الحواشي يمكن أن تتم بطريقتين: باستعمال ال reflection أو باستعمال الآداة apt.

استعمال ال reflection

الحواشي التي يمكن معالجتها باستعمال الreflection هي تلك التي يتم حفظها إلى وقت التشغيل: أي أننا أعطينا القيمة RetentionPolicy.RUNTIME إلى Retention@.

معالجة الحواشي باستعمال الreflection تعتمد على الواجهة java.lang.reflect.AnnotatedElement.

الفئات التالية هي التي تقوم بتطبيق هذه الواجهة: Class, Constructor, Field, Method, Package .

الواجهة AnnotatedElement تُعلن أربع طرق:

```
public boolean isAnnotationPresent(Class<? extends Annotation> annotationClass)
```

هذه الطريقة تعيد true إذا كانت الحاشية المُمَرَّرة لها (المعطى annotationClass) مُطَبقة على العنصر الذي ينادي على هذه الطريقة (أقصد بالعنصر، إما فئة، constructor، حقل، طريقة أو حزمة)، و false إن لم تكن الحاشية موجودة. هذه الطريقة مفيدة خاصة بالنسبة لل marker annotations .

```
<T extends Annotation> T getAnnotation(Class<T> annotationClass)
```

هذه الطريقة تنتظر نوع الحاشية التي نريد الحصول عليها، ثم تعيد لنا الحاشية إذا كانت موجودة، أو null في الحالة المعاكسة.

```
Annotation[] getAnnotations()
```

تعيد هذه الطريقة مجموع الحواشي الموجودة، على شكل مصفوفة. إن لم يكن هناك أي حاشية، فإن الطريقة تعيد مصفوفة فارغة.

```
Annotation[] getDeclaredAnnotations()
```

تعيد هذه الطريقة مجموع الحواشي الموجودة في العنصر، مع استثناء الحواشي الموروثة من الفئة الأم.

مثال.

لنفترض أننا نملك مجموعة من ال JavaBeans، ونريد إنشاء جداول في قواعد مطابقة لها في قاعدة البيانات.

أولا، نكتب الحاشية TableBean.

قلنا سابقا أن الreflection تستعمل فقط بالنسبة للحواشي التي نعطي فيها القيمة RetentionPolicy.RUNTIME إلى @Retention.

أيضا، هذه الحاشية ستطبق فقط على JavaBeans، أي الفئات، وبالتالي سنعطي القيمة ElementType.TYPE إلى @Target.

الحاشية فارغة، أي أنها marker annotation.

```java
import java.lang.annotation.ElementType;

import java.lang.annotation.Retention;

import java.lang.annotation.RetentionPolicy;

import java.lang.annotation.Target;


@Retention(RetentionPolicy.RUNTIME)

@Target(ElementType.TYPE)

public @interface TableBean {


}
```

الآن سنكتب الحواشي الخاصة بالحقول.

كل حقل في الJavaBean سيمثل عمودا في الجدول. كل عمود في الجدول له عدة خصائص (النوع، NULL, NOTNULL، primary key, foreign key, unique ...) لكن سنكتفي في المثال بتحديد النوع فقط.

سنكتب حاشيتين لنوعين فقط، Int و VarChar. الحاشية Int فارغة أيضا، حيث لسنا بحاجة لأي معلومات تخص الحقل، أما الحاشية VarChar فتمتلك الخاصية length، وهي تمثل طول النص.

طبعا، @Retention لها نفس قيمة الحاشية السابقة، أي RetentionPolicy.RUNTIME.

أما @Target، فستكون قيمتها ElementType.FIELD، أي أن الحواشي تنطبق على الحقول فقط.

```java
import java.lang.annotation.ElementType;

import java.lang.annotation.Retention;
```

```java
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.FIELD)
public @interface VarChar {
        int length() default 20;
}
```

```java
import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.FIELD)
public @interface Int {


}
```

الآن سنكتب الفئة التي ستقوم بمعالجة هذه الحواشي.

مبدأ المُعالج بسيط.

أولا نعطيه مصفوفة من الكائنات من فئة Class.

يقوم المعالج بالمناداة على الطريقة isAnnotationPresent(TableBean.class) لمعرفة إذا كانت الفئة تحتوي على الحاشية TableBean@.

إذا وجد الحاشية، فإنه يقوم باسترجاع كل الحقول الموجودة في الفئة.

بالنسبة لكل حقل، يقوم بالتحقق من وجود الحواشي Int@ أو VarChar@

إذا كانت موجودة، فإنه يخزن إسم الحقل، ونوعه (Int أو VarChar) في Hashtable.

بعد المرور على كل حقول الفئة، فإن المعالج ينشئ الإستعلام SQL .

```
import java.lang.reflect.Field;

import java.util.Hashtable;


public class AnnotationsProcessor {


        private Class[] beansToProcess;


        public AnnotationsProcessor(Class[] beansToProcess) {

                this.beansToProcess = beansToProcess;

        }


        public void process() {


                for (Class bean : beansToProcess) {


                        //A Hashtable containing the name of the column, and its
type

                        Hashtable<String, String> columns = new
Hashtable<String, String>();
```

```
                //A StringBuilder containing the result query
                StringBuilder query = new StringBuilder();

                if (isTableBean(bean)) {
                        //Get the table name
                        String tableName =
bean.getSimpleName().toLowerCase();
                        //Get all the fields
                        Field[] fields = bean.getDeclaredFields();

                        for (Field f : fields) {
                                if (isVarChar(f)) {
                                        //Get the length element from the
annotation
                                        int length =
f.getAnnotation(VarChar.class).length();
                                        columns.put(f.getName(),
"VARCHAR(" + length + ")");
                                } else if (isInt(f)) {
                                        columns.put(f.getName(), "INT");
                                }
                        }

                        query.append("CREATE TABLE " + tableName + "(");
                        for (String columnName : columns.keySet()) {
                                query.append(columnName + " " +
columns.get(columnName) + ",");
                        }
```

```
                                query.deleteCharAt(query.lastIndexOf(","));
                                query.append(");");
                                System.out.println(query.toString());
                                /*
                  * Execute the query with JDBC
                  */
                        }
                }
        }

        private boolean isTableBean(Class c) {
                return c.isAnnotationPresent(TableBean.class);
        }

        private boolean isVarChar(Field f) {
                return f.isAnnotationPresent(VarChar.class);
        }

        private boolean isInt(Field f) {
                return f.isAnnotationPresent(Int.class);
        }
}
```

JavaBeans بسيط

```java
@TableBean
public class Person {

	@VarChar(length=40) private String name;
	@Int private int age;

	public int getAge() {
		return age;
	}

	public void setAge(int age) {
		this.age = age;
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}
}
```

ثم الـ main

الشغل بيكون في برنامج NetBeans او اي برنامج آخر بالاضافة الى برنامج الاكسس

```
public class Main {


        public static void main(String[] args) {

                Class[] c = {Person.class};

        AnnotationsProcessor proc = new AnnotationsProcessor(c);

                proc.process();

        }

}
```

المثال بسيط جدا، ولا يجب الإعتماد عليه للحكم على فائدة الحواشي من عدمها.

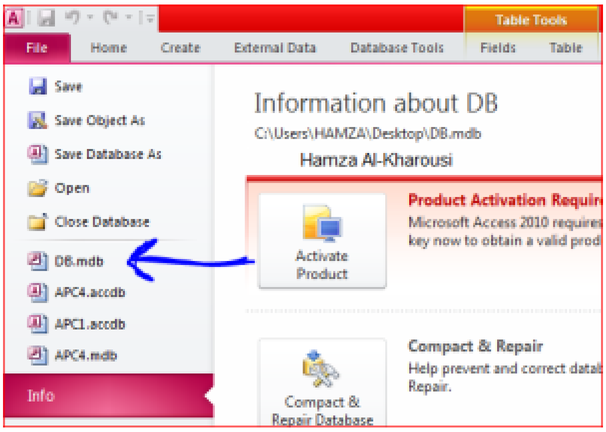
Databases

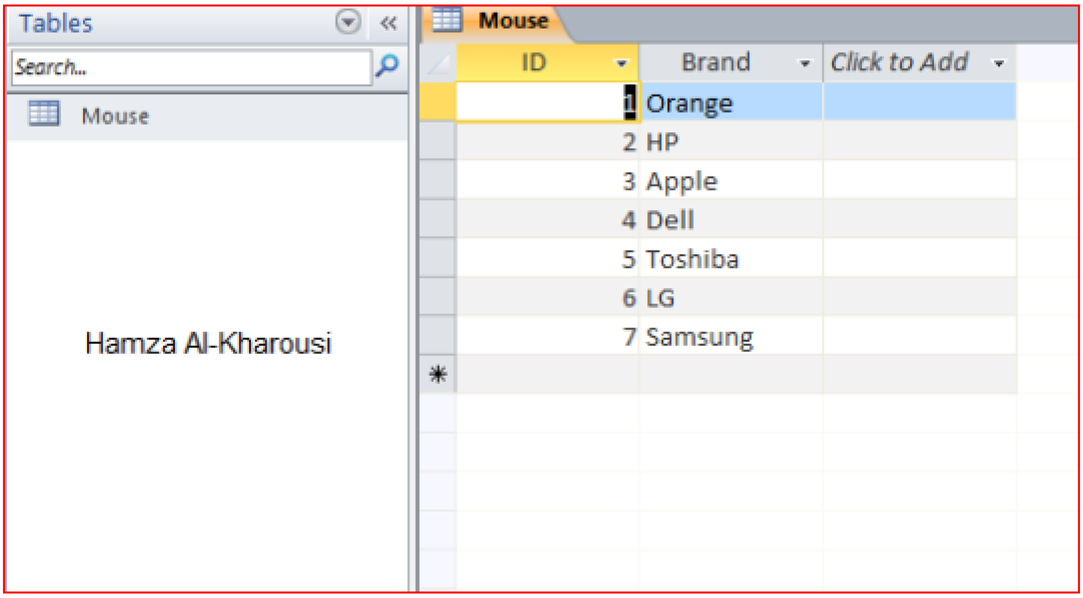
س ٧٢: كيفية ربط قاعدة بيانات اكسس(access) بالجافا ؟

الشغل بيكون في برنامج NetBeans او اي برنامج آخر بالاضافة الى برنامج الاكسس

.... نبدأ على بركة الله ....

نفرض ان عندنا قاعدة بيانات باسم DB وفيها جدول باسم Mouses



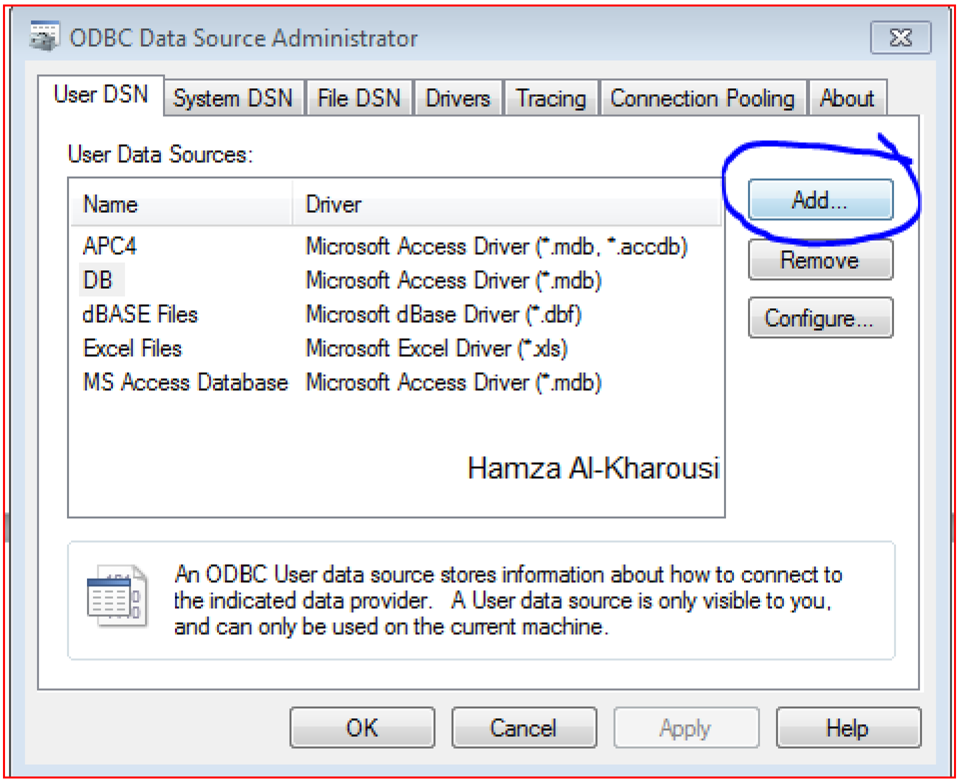


الآن نفتح لوحة التحكم Control Panel من زر أبدا

بعد ذلك نذهب الى System and Security

ثم نختار Administrative tools

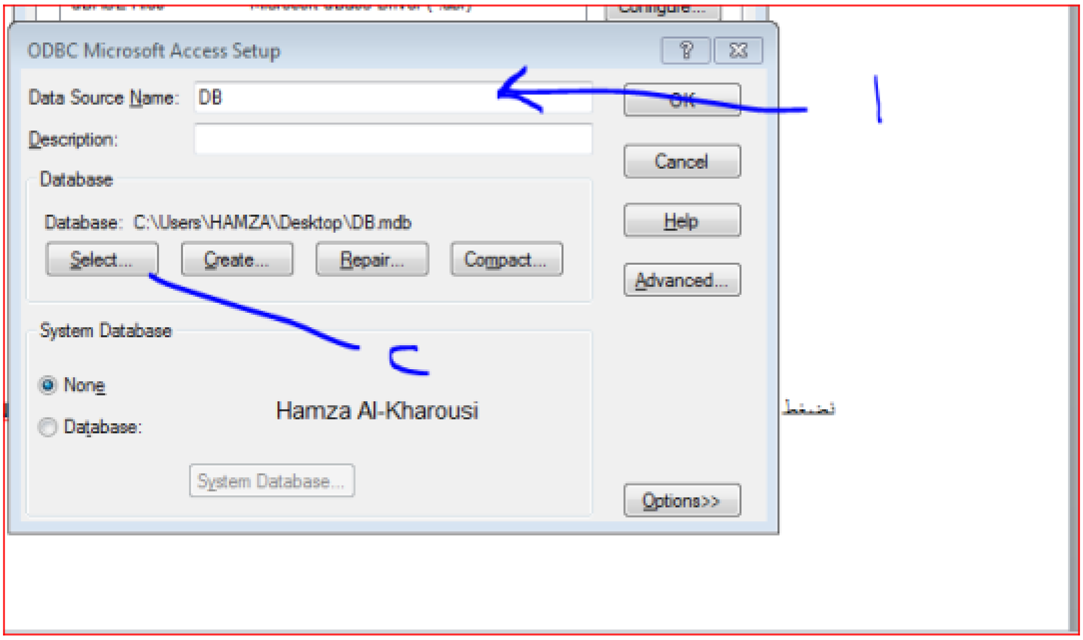
ونفتح (Data Source (ODBC



نضغط على زر add لكي نضيف قاعدة البيانات الي النظام ODBC ويتم التعرف عليها من خلال NetBeans

بعد الضغط على زر add

نقوم باختيار (Microsoft Access Driver (*.mdb, *accdb



١- نقوم بكتابة اسم قاعدة البيانات والذي سيكون كرابط لقاعدة البيانات (الاسم يكون محفوظ في ODBC ولا يشترط ان يكون نفس اسم قاعدة البيانات الاصلي)

٢- نقوم بتحديد مكان قاعدة البيانات

بعد ذلك نضغط ok

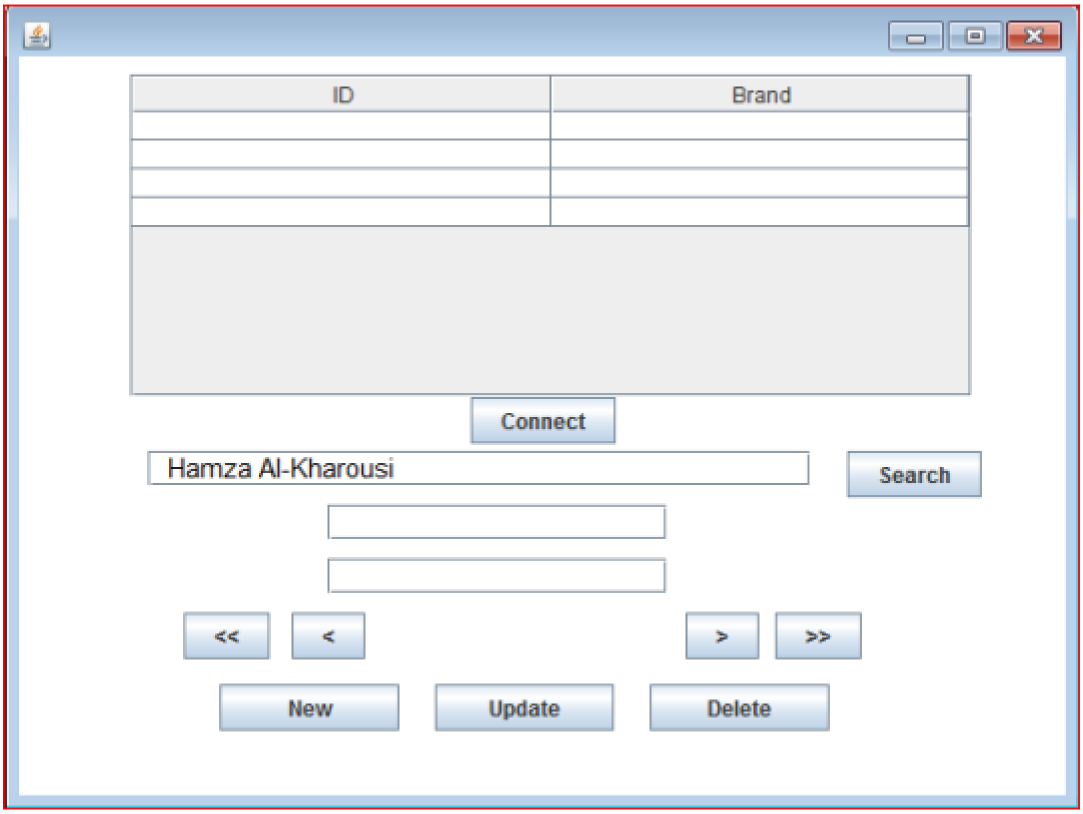
الان تم اضافة قاعدة البيانات الى ODBC

....

نذهب الى البرنامج الذي نريد توصيله بقاعدة البيانات

اولا : طريقة استيراد جدول واظهاره كاملا

نقوم بتجهيز الفريم التالي



ننتقل الى وضعSource

ونقوم بإستدعاء المكتبة الخاصة بالتعامل مع SQL

Import java.sql*؛

بعد ذلك نقوم باضافة الدالة (Method) التالية :

```java
19      private Connection con;
20      private Statement sql;
21      private String Search;
22      private ResultSet rs;
23
24
25  private void connDB(){//Hamza Al-Kharousi
26          try {
27                  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
28                   con=DriverManager.getConnection("jdbc:odbc:DB");
29                   sql=con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDATABLE);
30                   rs=sql.executeQuery("SELECT * FROM HOUSE" + Search );
31                   JOptionPane.showMessageDialog(null, "Connected");
32                   DefaultTableModel dtm= new DefaultTableModel();
33                   jTable1.setModel(dtm);
34                  dtm.addColumn("ID");
35                  dtm.addColumn("Brand");
36                  if(rs.next()){
37                      jTextField1.setText(rs.getString("ID"));
38                      jTextField2.setText(rs.getString("Brand"));
39                      rs.first();
40                      do{
41                          Object ary[]={rs.getInt("ID"), rs.getString("Brand")};
42                          dtm.addRow(ary);
43
44                      }while(rs.next());
45                      rs.first();
46                  }
47                  else{
48                      jTextField1.setText("");
49                      jTextField2.setText("");
50                      jTable1.removeAll();
51
52            }
53             }
           catch (Exception ex) {
55                  JOptionPane.showMessageDialog(null, ex.getMessage());
56                  System.exit(0);
57             }
58  }
```


سوف نقوم بشرح كل سطر :

١٩ – نقوم بتعريف المتغير con ليقوم بربط قاعدة البيانات التي انشأناها

٢٠- نقوم بتعريف المتغير sql ليقوم بتنفيذ جمل الاستعلام

٢١ – نقوم بتعريف المتغير search وهذا سيتم شرحه لاحقا

٢٢- نعرف المتغير rs لحفظ نتايج الاستعلام (البيانات)

٢٥- نقوم بانشاء الدالة connDB

٢٦ – نستخدم try – catch للتحذير من وجود اي خطأ

٢٧- Class هي مكتبة خاصة بلغة الجافا و forName هي خاصية تابعة في الكلاس نستخدمها لتقوم باستدعاء JBDC لتتم عملية ربط الجافا بقاعدة البيانات

٢٨ - الاتصال بقاعدة البيانات عن طريق الرابط DB الذي انشأناه

٢٩- انشاء جمل الاستعلام وتطبيقها داخل المشروع

٣٠- تنفيذ جمل الاستعلام المطلوبة

٣١- تقوم بانشاء نافذة لكي تتأكد ان الاتصال قد تم بنجاح

٣٢ – نعرف مخزن للجدول وحيث ان الجدول لا يتم تعبئته الا عن طريق مصفوفه والمصفوفة تأخذ البيانات من DB وتقوم بتعبئة الجدول

٣٣- نجعل الجدول يأخذ بياناته من dtm

٣٤ &٣٥– نضيف عمودين ف dtm والذي سينقل البيانات الى الجدول ونسمي الاول ID والثاني Brand

٣٦ – بعد جلب البيانات من DB وحفظها في rs اذا كان هنالك سطر تالي نفذ الاتي

٣٩ – يقوم بالبدأ من الصف الاول

٤٠ -٤٤ – يقوم بإنشاء لوب لادخال البيانات في dtm

٤٥- يقوم بالرجوع الى الصف الاول

٤٧-٥٢ – اذا كانت DB خالية فسيتم ابقاء الحقول المحددة خالية

٥٥ – في حال وجود خطأ ، سيتم اظهار رسالة مبينة الخطأ

٥٦ – الخروج من البرنامج

كود الزر Connect

عند الضغط عليه يتم استدعاء الدالة connDB وبدأ الاتصال بقاعدة البيانات

```
private void btnconnectActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Search="";
     connDB();                Hamza Al-Kharousi

}
```

كود الزر Search

في حالة ان المستخدم يريد البحث في الجدول يقوم بالكتابة في الحقل JTextField3 وتنفيذ المتغير Search

اما اذا كان الحقل فارغ فإن المتغير Search يكون فارغا

```java
private void btnSearchActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if(jTextField3.getText().equals("")){
        Search="";
        connDB();
    }
    else{
        //Don't forget to leave space before where
       Search = " where Brand like '%"+ jTextField3.getText()+"%'";
        connDB();
    }
}
```


كود الزر >

لعرض الصف السابق

```java
private void btnPreviousActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
   try{
    if(rs.previous()){

        jTextField1.setText(rs.getString("ID"));
        jTextField2.setText(rs.getString("Brand"));

    }
    }
       catch(Exception ex){
        JOptionPane.showMessageDialog(null, ex.getMessage());
    }

}
```

كود الزر >>

لعرض آخر صف في الجدول

```java
private void btnLastActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
        try{
      if(rs.last()){
          jTextField1.setText(rs.getString("ID"));
          jTextField2.setText(rs.getString("Brand"));

      }
    }
    catch(Exception ex){
        JOptionPane.showMessageDialog(null, ex.getMessage());
    }
}
```


كود زر <

لعرض الصف التالي

```java
private void btnNextActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
   try{
    if(!(rs.isLast())&& rs.next()){
         jTextField1.setText(rs.getString("ID"));
         jTextField2.setText(rs.getString("Brand"));


    }
   }
   catch(Exception ex){
     JOptionPane.showMessageDialog(null, ex.getMessage());

   }
}
```


كود زر >>

لعرض أول عمود في الجدول

```
    private void btnFirstActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
try{
        if(rs.first()){                        Hamza Al-Kharousi

            jTextField1.setText(rs.getString("ID"));
            jTextField2.setText(rs.getString("Brand"));

        }
        }
           catch(Exception ex){
            JOptionPane.showMessageDialog(null, ex.getMessage());
        }

    }
```

كود زر New

لإضافة عمود جديد

```
    private void btnNewActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
if (btnNew.getText().equals("New")){
    jTextField1.setEnabled(true);
    jTextField2.setEnabled(true);
    jTextField1.setText("");
    jTextField2.setText("");
    btnNew.setText("Save");
}
else{ if(!(jTextField1.getText().equals(""))&& !(jTextField2.getText().equals(""))){
    try{
        String s ="insert into mouse values(?,?)";
        PreparedStatement ps= con.prepareStatement(s);
        ps.setInt(1,Integer.parseInt(jTextField1.getText()));
        ps.setString(2, jTextField2.getText());
        int t = ps.executeUpdate();
        if(t>0){
            JOptionPane.showMessageDialog(null, "Saved....");
            connDB();
            jTextField1.setEnabled(false);
            jTextField2.setEnabled(false);
        }
         btnNew.setText("New");
    }
    catch(Exception ex){
        JOptionPane.showMessageDialog(null, ex.getMessage());


        }


}
```

Hamza Al-Kharousi

كود زر Update

لعمل تغير في بيانات صف من الجدول

```
private void btnUpdateActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        if (btnUpdate.getText().equals("Update")){
            jTextField1.setEnabled(true);
            jTextField2.setEnabled(true);
            btnUpdate.setText("Save");
        }
        else{
            try{
                String s = "Update mouse set Brand = ? "+ "where ID = ?";
                PreparedStatement ps = con.prepareStatement(s);
                ps.setString(1, jTextField2.getText());
                ps.setInt(2,Integer.parseInt(jTextField1.getText()));
               int t = ps.executeUpdate();
                if(t>0){
                    JOptionPane.showMessageDialog(null, "Updated...");
                    connDB();

                }

            }
            catch(Exception ex){
        JOptionPane.showMessageDialog(null, ex.getMessage());
            }

        }
    }
```

Hamza Al-Kharousi

كود زر Delete

لمسح اي صف من صفوف الجدول

```
private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
        if(jTextField1.getText().equals("")){
            JOptionPane.showConfirmDialog(null, "Can't Delete Field");
        }
        else{
            try{
            String s = "Delete from Mouse where ID = ?";
            PreparedStatement ps =con.prepareStatement(s);
            ps.setInt(1,Integer.parseInt(jTextField1.getText()));
            int t =ps.executeUpdate();
            if(t>0){
                JOptionPane.showMessageDialog(null, "Delleted....");
            connDB();

            }
            }
        catch(Exception ex){
          JOptionPane.showMessageDialog(null, ex.getMessage());

        }

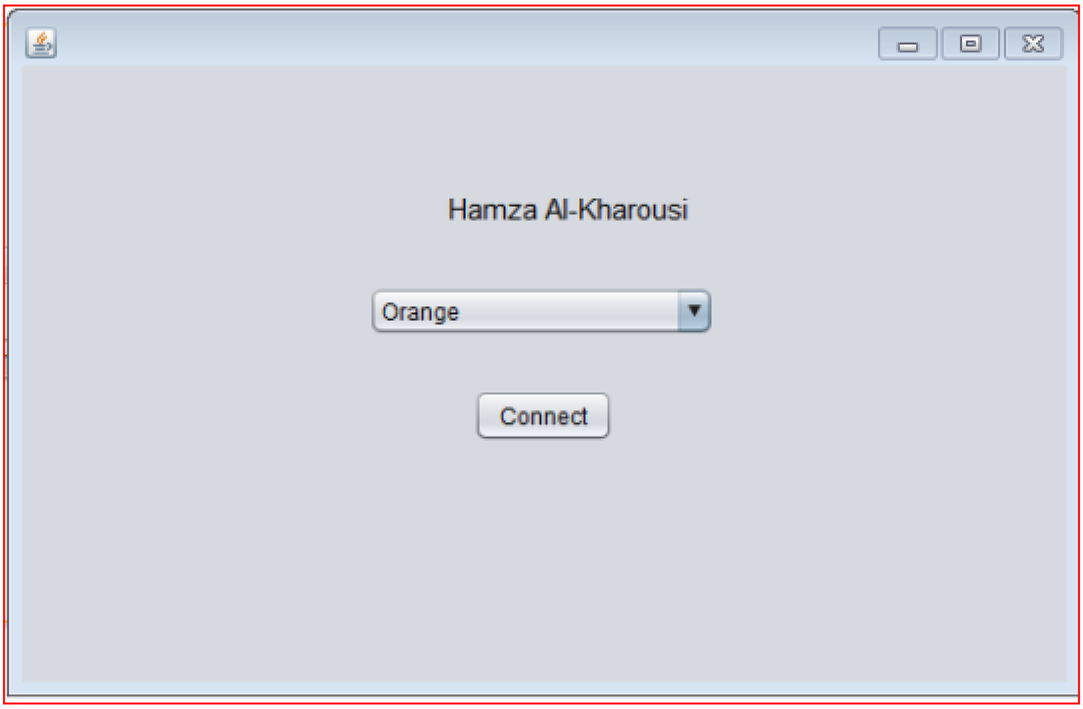
        }
    }
```

Hamza Al-Kharousi

ثانيا : طريقة استيراد عمود ووضعه محتواه في كومبوبوكس

نقوم بتجهيز الفريم التالي

وأحببت ان يكون بسيط



ستكون الدالة connDB بهذه الطريقة ،،،

```
        public Connection con;
         public Statement sql;
          public ResultSet rs;
      private void connDB(){ //* @author HAMZA

          try {

              Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
              con=DriverManager.getConnection("jdbc:odbc:DB");
              sql=con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDATABLE);

              rs = sql.executeQuery("SELECT Brand FROM Mouse");


             JOptionPane.showMessageDialog(null, "Database has been Connected");

              rs.first();


                  //--------------------------------
                do{
                    MouseComboBox.addItem(rs.getString("BRAND"));
 }while(  rs.next() );


               }
           catch (Exception ex) {
                   JOptionPane.showMessageDialog(null, ex.getMessage());
                   System.exit(0);
               }


     }
```

في هذه الدالة قمت بتغير قيمة المتغير rs حيث سيقوم بجلب فقط عمود واحد فقط وهو BRAND

وقمت باستخدام do – while لوب لتعبئة الكومبو بوكس

Graphics 2D

س ٧٣: كيف أرسم أشكال هندسية في الجافا؟

نقوم بالرسم في أحد مكونات Swing غالبا في JPanel. وكل عمليات الرسم تتم داخل الطريقة .paintComponent

هذه الطريقة تنتظر كائنا من فئة Graphics، هذا الكائن يمكن اعتباره فرشاة الرسم، فهو الذي يوفر لنا الطرق التي تمكننا من الرسم وتغيير لون الرسم .

من بين الطرق(الدوال) التي تحتوي عيها الفئة Graphics

```
drawRect(int x, int y, int width, int height)
```

هذه الطريقة ترسم مستطيلا فارغا. x و y يمثلان الإحداثيات التي سنبدأ منها الرسم، width و height يمثلان العرض والطول.

```
fillRect(int x, int y, int width, int height)
```

هذه الطريقة ترسم مستطيلا مملوءً. x و y يمثلان الإحداثيات التي سنبدأ منها الرسم، width و height يمثلان العرض والطول.

```
drawOval(int x, int y, int width, int height)
```

هذه الطريقة ترسم دائرة فارغة. x و y يمثلان الإحداثيات التي سنبدأ منها الرسم، width و height يمثلان العرض والطول.

```
fillOval(int x, int y, int width, int height)
```

هذه الطريقة ترسم دائرة مملوءة. x و y يمثلان الإحداثيات التي سنبدأ منها الرسم، width و height يمثلان العرض والطول.

```
drawLine(int x1, int y1, int x2, int y2)
```

هذه الطريقة ترسم سطرا. x1 و y1 يمثلان إحداثيات بداية الخط، x2 و y2 يمثلان إحداثيات النهاية.

```
setColor(Color c)
```

هذه الطريقة تحدد اللون الذي سيتم استعماله في الرسم.

هذه فقط بعض الطرق الموجودة في الفئة Graphics، لمعرفة كل الإمكانيات التي تتيحها هذه الفئة، يمكنكم مراجعة توثيق Sun.

مثال

```
import java.awt.Color;

import java.awt.Graphics;

import javax.swing.JFrame;

import javax.swing.JPanel;


public class TestPaint {


        public static void main(String[] args) {

                JFrame frm = new JFrame();

                frm.add(new PaintBoard());

                frm.setSize(300, 400);

                frm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

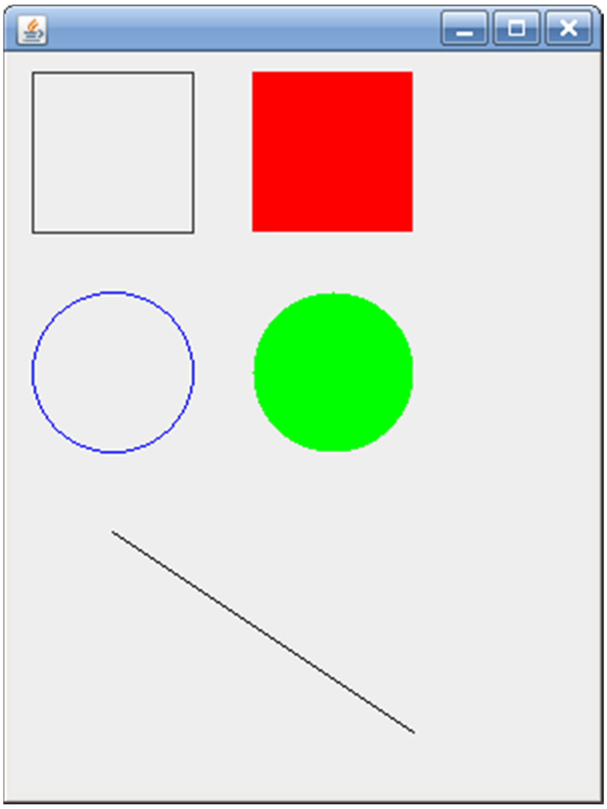
                frm.setVisible(true);
```

```java
        }
}

class PaintBoard extends JPanel {

        @Override
        protected void paintComponent(Graphics g) {
                super.paintComponent(g);
                g.drawRect(10, 10, 80, 80);
                g.setColor(Color.RED);
                g.fillRect(120, 10, 80, 80);
                g.setColor(Color.BLUE);
                g.drawOval(10, 120, 80, 80);
                g.setColor(Color.GREEN);
                g.fillOval(120, 120, 80, 80);
                g.setColor(Color.BLACK);
                g.drawLine(50, 240, 200, 340);
        }
}
```

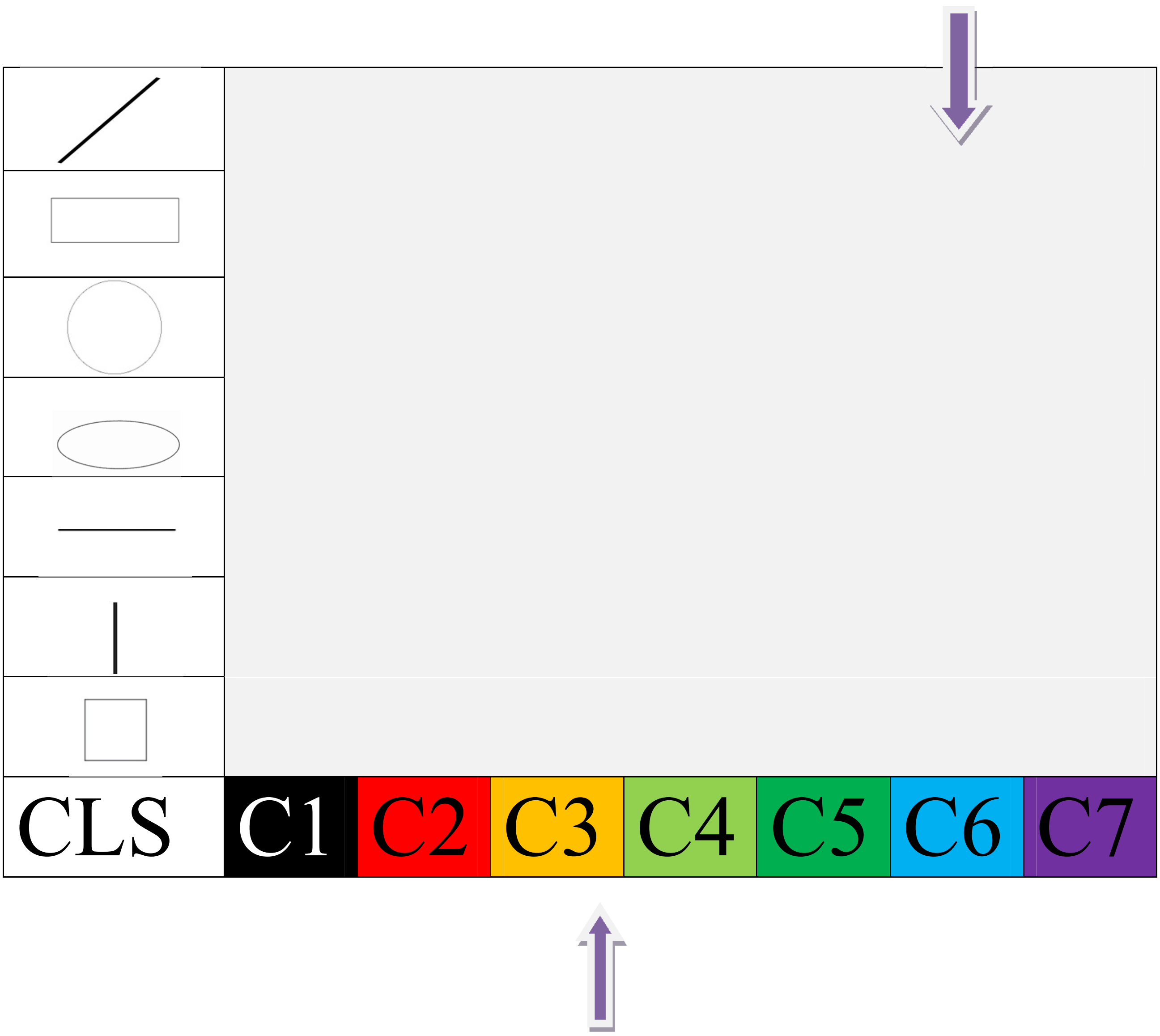
والنتيجة



مشاريع جاهزة

٧٤ ـ عمل برنامج شبيه بـ "الرسام". يقوم المستخدم بتحديد اللون و الشكل وينقر على الشاشة ليبدأ بالرسم ، استخدام شرح تقنيات البناء في الصف لرسم الشكل المحدد انظر للشكل في الأسفل.

(الزر CLS) لمسح منطقة عمل .



Colors

```
//Main :
package cgproject_4;
public class CGProject_4 {
    public static void main(String[] args) {

        Paint P=new Paint();
        P.setDefaultCloseOperation(Paint.EXIT_ON_CLOSE);
        P.setVisible(true);
        }
}

//Classe:

package cgproject_4;

import java.awt.Color;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import javax.swing.JButton;
```

```
import javax.swing.JFrame;

import javax.swing.JPanel;




public class Paint extends JFrame implements ActionListener,MouseListener

{

    private int x1,y1,x2,y2;

   private Color c;

   private JPanel jpanel;

   private JButton
bt1,bt2,bt3,bt4,bt5,bt6,bt7,bt8,bt9,bt10,bt11,bt12,bt13,bt14,bt15,bt16,bt17,bt1
8;

   private Graphics2D g;

   private int count=0,ch;

   private float r,a,b;


     Paint()

     {

      super("Paint : Hekam");

      setSize(900,680);


          this.setLayout(null);


          bt1=new JButton(){


          @Override
```

```
protected void paintComponent(Graphics g){
super.paintComponent(g);
Graphics2D g2d = (Graphics2D) g;

   g2d.setColor(Color.BLACK);

   g2d.drawLine(20,60,60,20);
}

};
bt1.setLocation(0,0);
bt1.setSize(80,80);
bt1.addActionListener(this);
this.add(bt1);

bt2=new JButton(){
@Override
protected void paintComponent(Graphics g) {
    super.paintComponent(g);
    Graphics2D g2d = (Graphics2D) g;
    g2d.setColor(Color.BLACK);
    g2d.drawRect(15,15,50,40);
}

};
bt2.setLocation(0,80);
```

```
bt2.setSize(80,80);
bt2.addActionListener(this);

this.add(bt2);

bt3=new JButton(){

@Override
protected void paintComponent(Graphics g) {
    super.paintComponent(g);
    Graphics2D g2d = (Graphics2D) g;
    g2d.setColor(Color.BLACK);
    g2d.drawOval(15,15,50,50);
}
};
bt3.setLocation(0,160);
bt3.setSize(80,80);
bt3.addActionListener(this);
this.add(bt3);

bt4=new JButton(){

@Override
protected void paintComponent(Graphics g) {
    super.paintComponent(g);
    Graphics2D g2d = (Graphics2D) g;
```

```
        g2d.setColor(Color.BLACK);
        g2d.drawOval(10,15,60,50);
    }
    };
    bt4.setLocation(0,240);
    bt4.setSize(80,80);
    bt4.addActionListener(this);
    this.add(bt4);

    bt5=new JButton(){

    @Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
        Graphics2D g2d = (Graphics2D) g;
        g2d.setColor(Color.BLACK);
        g2d.drawLine(40,10,40,65);
    }
    };
    bt5.setLocation(0,320);
    bt5.setSize(80,80);
    bt5.addActionListener(this);
    this.add(bt5);


    bt6=new JButton(){
```

```
@Override
protected void paintComponent(Graphics g) {
    super.paintComponent(g);
    Graphics2D g2d = (Graphics2D) g;
    g2d.setColor(Color.BLACK);
    g2d.drawLine(10,40,60,40);
}
};
bt6.setLocation(0,400);
bt6.setSize(80,80);
bt6.addActionListener(this);
this.add(bt6);

bt7=new JButton(){

@Override
protected void paintComponent(Graphics g) {
    super.paintComponent(g);
    Graphics2D g2d = (Graphics2D) g;
    g2d.setColor(Color.BLACK);
    g2d.drawRect(15,15,45,45);
}
};
bt7.setLocation(0,480);
bt7.setSize(80,80);
```

```
bt7.addActionListener(this);
this.add(bt7);

bt8=new JButton("CLS");
bt8.setLocation(0,560);
bt8.setSize(80,80);
bt8.addActionListener(this);
this.add(bt8);




bt9=new JButton();
bt9.setBackground(Color.black);
bt9.setLocation(80,560);
bt9.setSize(80,80);
bt9.addActionListener(this);
this.add(bt9);

bt10=new JButton();
bt10.setBackground(Color.red);
bt10.setLocation(160,560);
bt10.setSize(80,80);
bt10.addActionListener(this);
this.add(bt10);
```

```
bt11=new JButton();
bt11.setBackground(Color.yellow);
bt11.setLocation(240,560);
bt11.setSize(80,80);
bt11.addActionListener(this);
this.add(bt11);

bt12=new JButton();
bt12.setBackground(Color.orange);
bt12.setLocation(320,560);
bt12.setSize(80,80);
bt12.addActionListener(this);
this.add(bt12);

bt13=new JButton();
bt13.setBackground(Color.green);
bt13.setLocation(400,560);
bt13.setSize(80,80);
bt13.addActionListener(this);
this.add(bt13);

bt14=new JButton();
bt14.setBackground(Color.cyan);
bt14.setLocation(480,560);
bt14.setSize(80,80);
bt14.addActionListener(this);
```

```
this.add(bt14);


bt15=new JButton();

bt15.setBackground(Color.magenta);

bt15.setLocation(560,560);

bt15.setSize(80,80);

bt15.addActionListener(this);

this.add(bt15);


bt16=new JButton();

bt16.setBackground(Color.white);

bt16.setLocation(640,560);

bt16.setSize(80,80);

bt16.addActionListener(this);

this.add(bt16);


bt17=new JButton();

bt17.setBackground(Color.darkGray);

bt17.setLocation(720,560);

bt17.setSize(80,80);

bt17.addActionListener(this);

this.add(bt17);


bt18=new JButton("Exit");

bt18.setSize(80,80);

bt18.setLocation(800,560);
```

```
          bt18.addActionListener(this);
           this.add(bt18);

          jpanel=new JPanel();

          jpanel.setLocation(80,0);
          jpanel.setSize(815,560);
          jpanel.setBackground(Color.white);
          jpanel.addMouseListener(this);
          this.add(jpanel);



}
     public void setColor(Color color){
          c =color;
          }
    public  void BCircle(Graphics g, int xc, int yc, int r){
        int x = 0, y = r, s = 3-(2*r);

        while(x<=y){
        g.drawLine(xc+x, yc+y, xc+x, yc+y);
        g.drawLine(xc+x, yc-y, xc+x, yc-y);
        g.drawLine(xc-x, yc+y, xc-x, yc+y);
        g.drawLine(xc-x, yc-y, xc-x, yc-y);
        g.drawLine(xc+y, yc+x, xc+y, yc+x);
```

```
        g.drawLine(xc+y, yc-x, xc+y, yc-x);
        g.drawLine(xc-y, yc+x, xc-y, yc+x);
        g.drawLine(xc-y, yc-x, xc-y, yc-x);
            if (s<0)
                s += 4*x + 6;
            else{
                s += 4*(x-y) + 10;
                y--;
            }
            x++;
        }
         }
    public void Ellipse(Graphics g,int xc,int yc,int a,int b){
        double ct,st;
        double xt;
        double x,y;

        double theta=2.0/(a+b);

        ct=Math.cos(theta);
        st=Math.sin(theta);
        x=0;        y=b;



        while (y>=0)
```

```
        {

            PutPixel(g,(int)Math.round(x+xc),(int)Math.round(yc-y));

            PutPixel(g,(int)Math.round(xc-x),(int)Math.round(yc-y));

            PutPixel(g,(int)Math.round(xc-x),(int)Math.round(yc+y));

            PutPixel(g,(int)Math.round(xc+x),(int)Math.round(yc+y));




            xt=x;

            x=(x*ct)+(a*1.0/b)*(y*st);


            y=(y*ct)-(b*1.0/a)*(x*st);


        }


    }


     public void PutPixel(Graphics g,int x,int y){

        g.drawLine(x, y, x, y);

    }

   @Override

public void actionPerformed(ActionEvent e) {

     if(e.getSource()==bt1){

         ch=1;

     }

     else if(e.getSource()==bt2){

         ch=2;
```

```
}
else if(e.getSource()==bt3){
    ch=3;
}
else if(e.getSource()==bt4){
    ch=4;
}
else if(e.getSource()==bt5){
    ch=5;
}
else if(e.getSource()==bt6){
    ch=6;
}
else if(e.getSource()==bt7){
    ch=7;
}
else if(e.getSource()==bt8){
    repaint();
}
else if(e.getSource()==bt9){
    setColor(Color.black);
}
else if(e.getSource()==bt10){
     setColor(Color.red);
}
else if(e.getSource()==bt11){
```

```
            setColor(Color.yellow);
        }
        else if(e.getSource()==bt12){
            setColor(Color.orange);
        }
        else if(e.getSource()==bt13){
            setColor(Color.green);
        }
        else if(e.getSource()==bt14){
            setColor(Color.cyan);
        }
        else if(e.getSource()==bt15){
            setColor(Color.magenta);
        }
        else if(e.getSource()==bt16){
            setColor(Color.white);
        }
        else if(e.getSource()==bt17){
            setColor(Color.lightGray);
        }

        else if(e.getSource()==bt18)
        {
            System.exit(0);
        }
    }
```

```
@Override
  public void mouseClicked(MouseEvent e) {
          count++;
      if(count==1){
     x1=e.getX();
     y1=e.getY();
      }
      else if(count==2){

      x2=e.getX();
      y2=e.getY();

      paintComponent(g);
      }




  }

  @Override
  public void mouseEntered(MouseEvent e) {
  }
  @Override
  public void mouseExited(MouseEvent e) {
```

```
    }
    public void mousePressed(MouseEvent e) {
    }
    public void mouseReleased(MouseEvent e) {
    }



     public void paintComponent(Graphics g) {
         g=(Graphics2D) jpanel.getGraphics();

         if(ch==1){
         g.setColor(c);
         g.drawLine(x1, y1, x2, y2);
         x1=0;y1=0; x2=0;  y2=0;
         count=0;
         }
         if(ch==2){
             if(x1>x2){
               int temp ;
                temp= x2;
               x2 = x1;
               x1 = temp;
             }
               if(y1>y2){
                 int temp = y2;
```

```
                y2 = y1;

                y1 = temp;

         }

             g.setColor(c);

             g.drawRect(x1, y1,x2-x1,y2-y1);

              x1=0;y1=0; x2=0;  y2=0;

              count=0;

          }

          if(ch==3){


           r=(float) Math.sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1));

          g.setColor(c);

          BCircle(g,x1,y1,Math.round(r));

            x1=0;y1=0; x2=0;  y2=0;

              count=0;

          }


           if(ch==4){

               a=(float) Math.sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1));

               b=a/2;

               g.setColor(c);

             Ellipse(g,x1,y1,Math.round(a),Math.round(b));

               x1=0;y1=0; x2=0;  y2=0;

              count=0;

 }
```

```
      if(ch==5){

            g.setColor(c);

            g.drawLine(x1, y1, x1, y2);

           x1=0;y1=0; x2=0;  y2=0;

           count=0;

}


      if(ch==6){

            g.setColor(c);

            g.drawLine(x1, y1, x2, y1);

           x1=0;y1=0; x2=0;  y2=0;

           count=0;

}


      if(ch==7){

           if(x1>x2){

                  int temp ;

                   temp= x2;

                  x2 = x1;

                  x1 = temp;

              }

                  if(y1>y2){

                    int temp = y2;

                  y2 = y1;

                  y1 = temp;
```

```
        }


         g.setColor(c);

         g.drawRect(x1, y1,x2-x1,(x2-x1));

         x1=0;y1=0; x2=0;  y2=0;

         count=0;

}


     }

  }
```

capture-٧٥

كيف تعمل صوره للشاشة من البرنامج

علما ان الصورة سوف تحتفظ في مسار البرنامج باسم screen.jpg

```
/*كـيف تـعمل صوره لـلشاشه بـظغطه زر مـن خلال الـبرنـامـج
```

```
 * To change this license header, choose License Headers in Project
Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

package javaapplication2;

import java.awt.AWTException;
import java.awt.Image;
import java.awt.Rectangle;
import java.awt.Robot;
import java.awt.Toolkit;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import javax.imageio.ImageIO;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JOptionPane;

/**
 *
```

```
 * @author mohammed s
 */
public class JavaApplication2 {
   static JFrame f;
   static JButton captur;
   static JButton close;
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {



         f=new JFrame("capture image");
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        f.setSize(600,600);
            f.setVisible(true);
            captur=new JButton("capture");
            captur.setBounds(10,500,100,50);
            f.add(captur);
            captur.addActionListener(new ActionListener() {
                @Override
                public void actionPerformed(ActionEvent e) {
                try {
                    Robot r=new Robot();
  BufferedImage buffer=r.createScreenCapture(new
Rectangle(Toolkit.getDefaultToolkit().getScreenSize()));
```

```
                    try {

                        ImageIO.write(buffer,"JPG",new
File("screen.jpg"));


                        JOptionPane.showMessageDialog(null," تم الحفظ في
مسار البرنامج "+File.pathSeparator);

                    } catch (IOException ex) {

                        ex.printStackTrace();

                    }

                } catch (AWTException ex) {

                    ex.printStackTrace();

                }


                }

            });

        // TODO code application logic here

    }

}
```

splash-٧٦

وهي كيف تعمل يعني في مقدمه البرنامج صوره جاري التحميل وصوره معبره عن البرنامج شاشه مدتها خمس ثواني

```
// Splash.java

//

import javax.swing.*;
```

```
import java.awt.*;


 class Splash {

  public static void main(String[] args) {


    // Throw a nice little title page up on the screen first

    showSplash(3000);


    System.exit(0); // replace with application code!

  }


  // A simple little method to show a title screen in the

  // center of the screen for a given amount of time.

   public static void showSplash(int duration) {

   JWindow splash = new JWindow();

    JPanel content = (JPanel)splash.getContentPane();


    // set the window's bounds, centering the window

    splash.setBounds(200,100,455,500);


    // build the splash screen

    JLabel label = new JLabel(new ImageIcon("mm.png"));

    JLabel copyrt = new JLabel

      ("wait ...", JLabel.CENTER);

    copyrt.setFont(new Font("Sans-Serif", Font.BOLD, 40));

    copyrt.setForeground(Color.green);
```

```
    content.add(label, BorderLayout.CENTER);

    content.add(copyrt, BorderLayout.SOUTH);

    content.setBorder(BorderFactory.createLineBorder(Color.blue, 20));


    // display it

    splash.setVisible(true);


    // Wait a little while, maybe while loading resources

    try { Thread.sleep(duration); } catch (Exception e) {}


    splash.setVisible(false);

  }

}
```

open net-٧٧

كيف تفتح مستعرض النت من برنامجك google.com مثلا

```
import java.awt.Desktop;

import java.io.IOException;

import java.net.URI;

import java.net.URISyntaxException;

class mo{
```

```
    public static void main(String[] args) {

Desktop desktop= Desktop.getDesktop();

            try {
                //open the default browser using the method browse which take
an URI object representing the web page
                desktop.browse(new URI("http://google.com"));
            } catch (URISyntaxException ex) {
                ex.printStackTrace();
            } catch (IOException ex) {
                ex.printStackTrace();
            }

}}
```

count letter-٧٨

كيف تعمل برنامج يحسب لك عدد ظهور كل حرف من الحروف الانجليزيه بطريقه محترفه جدا

```
import javax.swing.JOptionPane;

  class CountEachLetter {
    /**  Main method */
```

```
    public static void main(String[] args) {
      // Prompt the user to enter a string
      String s = JOptionPane.showInputDialog("Enter a string:");

      // Invoke the countLetters method to count each letter
int[] counts = countLetters(s.toLowerCase());

// Declare and initialize output string
String output = "";

// Display results
for (int i = 0; i < counts.length; i++) {
if (counts[i] != 0)
output += (char)('a' + i) + " appears " +
counts[i] + ((counts[i] == 1) ? " time\n" : " times\n");
}

// Display the result
JOptionPane.showMessageDialog(null, output);
}

// Count each letter in the string
 static int[] countLetters(String s) {
int[] counts = new int[26];

for (int i = 0; i < s.length() ; i++) {
```

```
if (Character.isLetter(s.charAt(i)))
counts[s.charAt(i) - 'a']++;
}

    return counts;
}

}
```

draw time-٧٩

وهو اعتبره برنامج رهيب جدا

لإنه يعطيك حقلين حقل تدخل فيه الساعة وحقل تدخل فيه الدقائق وتضغط الزر وهو يروح يعمل لها رسمه على الفورم

ممكن تعطيه تايمر الساعه حق الويندوز وهو يظل يرسم على طول

```
import java.awt.* ;
import java.awt.event.* ;
import javax.swing.* ;
class MaFenetre extends JFrame implements ActionListener
{ public MaFenetre ()
{ setTitle ("PENDULE") ;
```

```
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setSize (400, 250) ;
Container contenu = getContentPane() ;
panControles = new JPanel() ;
contenu.add (panControles, "North") ;
saisieHeures = new JTextField (4) ;
panControles.add (saisieHeures) ;
etiqHeures = new JLabel (" Heures") ;
panControles.add (etiqHeures) ;
saisieMinutes = new JTextField (4) ;
panControles.add (saisieMinutes) ;
etiqMinutes = new JLabel (" Minutes") ;
panControles.add (etiqMinutes) ;
ok = new JButton ("Mise a l'heure") ;
panControles.add (ok) ;
ok.addActionListener (this) ;
panPendule = new PanPendule(this) ;
contenu.add (panPendule) ;
panPendule.setBackground (Color.yellow) ;
}
public int getMinutes ()
{ return minutes ;
}
public int getHeures ()
{ return heures ;
}
```

```
public void actionPerformed (ActionEvent e)
{ int h, m ; // pour les valeurs saisies
if (e.getSource() == ok)
{ try
{ String chHeures = saisieHeures.getText() ;
h = Integer.parseInt (chHeures) ;
}
catch (NumberFormatException ex)
{ h = -1 ; // on force une valeur invalide
saisieHeures.setText ("") ;
}
try
{ String chMinutes = saisieMinutes.getText() ;
m = Integer.parseInt (chMinutes) ;
}
catch (NumberFormatException ex)
{ m = -1 ; // on force une valeur invalide
saisieMinutes.setText ("") ;
}
// si les valeurs obtenues sont valides, on les place dans
// les champs heures et minutes et on force le dessin
// sinon, on replace les anciennes valeurs dans les champs texte
  repaint() ;
if ((h>=0) && (h<24) && (m>=0) && (m<60))
{ heures = h ; minutes = m ;
```

```
}
else
{ saisieMinutes.setText (""+minutes) ;
saisieHeures.setText (""+heures) ;
}
}
}
private JPanel panControles ;
private PanPendule panPendule ;
private JTextField saisieHeures, saisieMinutes ;
private JLabel etiqHeures , etiqMinutes ;
private JButton ok ;
private int minutes=0, heures=0 ;
}
class PanPendule extends JPanel
{ public PanPendule (MaFenetre fen)
{ this.fen = fen ;
}
public void paintComponent (Graphics g)
{ super.paintComponent(g) ;
// dessin du cercle
Dimension dim = getSize() ;
int largeur = dim.width, hauteur = dim.height ;
boolean panTropLarge = (largeur>hauteur) ;
int xCentre = largeur/2, yCentre = hauteur/2 ;
int rayon ;
```

```
if (panTropLarge) rayon = hauteur/2 - 2 ; else rayon = largeur/2 - 2 ;
g.drawOval (xCentre-rayon, yCentre-rayon, 2*rayon, 2*rayon) ;
// dessin grande aiguille
int minutes = fen.getMinutes() ;
double angle = Math.PI/2 * (1. - minutes/15.) ;
g.drawLine (xCentre, yCentre,
(int)(xCentre+rayon*Math.cos(angle)),
(int)(yCentre-rayon*Math.sin(angle))) ;
// dessin petite aiguille
int heures = fen.getHeures() ;
angle = Math.PI/2 * (1. - heures/3. - minutes/180.) ;
g.drawLine (xCentre, yCentre,
(int)(xCentre+rayon/2.*Math.cos(angle)),
(int)(yCentre-rayon/2.*Math.sin(angle))) ;
}
private MaFenetre fen ;
}
class Pendule
{ public static void main (String args[])
{ MaFenetre fen = new MaFenetre() ;
fen.setVisible(true) ;
}
}
```

وهو كيف تلون الخط وكيف تغير نوع الخط باستخدام الجافا

```
import java.awt.Color;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.Action;

import javax.swing.ButtonGroup;

import javax.swing.JButton;

import javax.swing.JCheckBox;

import javax.swing.JColorChooser;

import javax.swing.JFrame;

import javax.swing.JOptionPane;

import javax.swing.JScrollPane;

import javax.swing.JTextArea;

class mo implements ActionListener {


 public JFrame f;

 public JCheckBox c1,c2;

 public JButton b1;


    private Color color=Color.GRAY;


    private Component mo;

 JTextArea t;

 public static void main(String[] args) {

     mo n=new mo();
```

```
}
public mo()
{
    f=new JFrame();
    f.setVisible(true);
    f.setSize(300,300);
    f.setLayout(null);
    c1=new JCheckBox("bold");
    c2=new JCheckBox("italic");
    t=new JTextArea();
    c1.setBounds(10,20,100,30);
    c2.setBounds(10,50,100,30);
        JScrollPane s=new JScrollPane(t);
    s.setBounds(10,100,100,100);
ButtonGroup g=new ButtonGroup();
g.add(c1);
g.add(c2);
    f.add(c1);
    f.add(c2);
    f.add(s);
    b1=new JButton("ch");
    b1.setBounds(10,200,100,50);
    b1.addActionListener(this);
    f.add(b1);
    c1.addActionListener(this);
```

```
     c2.addActionListener(this);

 }


    public void actionPerformed(ActionEvent e) {

    if(e.getSource()==c1)

    {String messg=t.getText();

     t.setFont(new Font("",Font.BOLD,14));

     t.setText(messg);


    }

    if(e.getSource()==c2)

    {String messg=t.getText();

    t.setFont(new Font("",Font.ITALIC,14));

    t.setText(messg);}

    if(e.getSource()==b1)

    {

        color=JColorChooser.showDialog(t,"choss",color);

        t.setCaretColor(color);

        t.setColumns(10);

        t.setLineWrap(true);

       // t.setBackground(color);

    }

}}
```

٨١- calc

كيف تعمل اله حاسبه باستخدام الجافا

```
import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;


import java.awt.*;

import java.util.Currency;

import java.util.Set;

import javax.swing.*;

import javax.swing.JFrame;

import javax.swing.JOptionPane;

import javax.swing.JTextField;

 class calc implements ActionListener{

    public JFrame f;

    public JButton  p0,p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11,p12,p13,p14,p15;

    public JTextField t1,t2,t3;

    public Double num1,sum;


    private JButton p;

    private  JLabel l1;


    public java.lang.Double min;
```

```
  private Color red = Color.lightGray;


  public java.lang.Double mult;

  String op;

  private ImageIcon icon;

  public static void main(String[] args) {


      calc ali=new calc();

      ali.method();


  }




  public void method() {


f=new JFrame("calc");

  f.setVisible(true);

  f.setBounds(20,20,1000,900);

  f.setLayout(null);

  f.setTitle("الة حاسبة");


 // icon = new ImageIcon ("k1.jpg");

  //f.add(icon);

  l1=new JLabel(new ImageIcon("l.png"));

  l1.setBounds(1,1,1000,900);
```

```
f.add(l1);
      p15=new JButton(new ImageIcon("clear.png"));
p1=new JButton(new ImageIcon("1.png"));
p1.setSelected(true);
p2=new JButton(new ImageIcon("2.png"));
p3=new JButton(new ImageIcon("3.png"));
p4=new JButton(new ImageIcon("4.png"));
p5=new JButton(new ImageIcon("5.png"));
p6=new JButton(new ImageIcon("6.png"));
p7=new JButton(new ImageIcon("7.png"));
p8=new JButton(new ImageIcon("8.png"));
p9=new JButton(new ImageIcon("9.png"));
p10=new JButton(new ImageIcon("+.png"));
p11=new JButton(new ImageIcon("-.png"));
p12=new JButton(new ImageIcon("m.png"));
p13=new JButton(new ImageIcon("=.png"));


p1.setForeground(Color.red);
p1.setBackground(Color.blue);


t1=new JTextField();
Color color1 = Color.red;
t1.setFont(new Font("",Font.PLAIN,18));
  p15.addActionListener(this);
p1.addActionListener(this);
p2.addActionListener(this);
```

```
  p3.addActionListener(this);
  p4.addActionListener(this);
  p5.addActionListener(this);
  p6.addActionListener(this);
  p7.addActionListener(this);
  p8.addActionListener(this);
  p9.addActionListener(this);
  p10.addActionListener(this);
  p11.addActionListener(this);
  p12.addActionListener(this);
  p13.addActionListener(this);
  t1.setBounds(600,100,200,100);
  //ImageIcon image = new ImageIcon ("k1.jpg");

   p1.setBounds(20,100,100,100);
   p2.setBounds(200,100,100,100);
  p3.setBounds(380,100,100,100);

// p15.setBounds(450,100,60,20);
  p4.setBounds(20,250,100,100);
  p5.setBounds(200,250,100,100);
  p6.setBounds(380,250,100,100);
  p7.setBounds(20,400,100,100);
  p8.setBounds(200,400,100,100);
  p9.setBounds(380,400,100,100);
   p15.setBounds(550,400,100,100);
```

```
    p10.setBounds(20,550,100,100);

    p11.setBounds(200,550,100,100);

    p12.setBounds(380,550,100,100);

    p13.setBounds(550,550,100,100);

   // p14.setBounds(700,550,100,100);

    l1.add(t1);

 l1.add(p1);

 l1.add(p2);

 l1.add(p3);

 l1.add(p15);

    l1.add(p4);

    l1.add(p5);

    l1.add(p6);

    l1.add(p7);

    l1.add(p8);

    l1.add(p9);

    l1.add(p10);

    l1.add(p11);

    l1.add(p12);

    l1.add(p13);

    //f.add(p14);


    }

    public void actionPerformed(ActionEvent e) {

if(e.getSource()==p1)

{//t1.setFont(new Font("",Font.BOLD,18));
```

```
   t1.setText(t1.getText().concat("1"));

}

else

if(e.getSource()==p2)

{t1.setText(t1.getText().concat("2"));

    //t1.setText("2");

}


 if(e.getSource()==p3)

{t1.setText(t1.getText().concat("3"));

    //t1.setText("3");

}


    if(e.getSource()==p4)

{t1.setText(t1.getText().concat("4"));

   // t1.setText("4");

}

    if(e.getSource()==p5)

{t1.setText(t1.getText().concat("5"));

   // t1.setText("5");

}

    if(e.getSource()==p6)

{t1.setText(t1.getText().concat("6"));

  //  t1.setText("6");

}

    if(e.getSource()==p7)
```

```
{t1.setText(t1.getText().concat("7"));
    //t1.setText("7");
}
    if(e.getSource()==p8)
{t1.setText(t1.getText().concat("8"));
   // t1.setText("8");
}
    if(e.getSource()==p9)
{t1.setText(t1.getText().concat("9"));
   // t1.setText("9");
}
    if(e.getSource()==p10)
{num1 = Double.parseDouble(t1.getText());
    t1.setText("");
    op="+";



}
    if(e.getSource()==p11)
{min = Double.parseDouble(t1.getText());
    t1.setText("");
    op="-";



}
if(e.getSource()==p15)
```

```
{num1=0.0;sum=0.0;
 mult=0.0;
    t1.setText("");
}
    if(e.getSource()==p12)
{num1 = Double.parseDouble(t1.getText());
    t1.setText("");
        op="*";


}
    if(e.getSource()==p13)
{if(op=="+"){
    sum= Double.parseDouble(t1.getText());
 sum+=num1;
    t1.setText(sum.toString());}
 if(op=="-")
 {  num1= Double.parseDouble(t1.getText());
  min=min-num1;
  t1.setText(min.toString());}
if(op=="*")
{mult= Double.parseDouble(t1.getText());
     mult*=num1;
      t1.setText(mult.toString());
}}}
}
```



أسماء المشاركين في الكتاب :

عبد اللطيف عبد العزيز محمد جامع

maisam alrawi

ميادة حمود عوض حسن

محمد امين

Mai Kamel Amro

حكم محمود عبدالله احميدات

Mostafa Anter

Sabeel Akhras

رامي عبدالكريم محمد الحمادي

حمزة براهمي

باسل سعيد

محمد سنان محسن مسعود

راكان فاضل

تم بحمد الله ☺